Este texto é a tradução para português da secção 2.4 (pp. 64-96) da tese doutoral de Leonel Morgado:

Morgado, Leonel (2005). *Framework for computer programming in preschool and kindergarten*. Tese de doutoramento. Vila Real, Portugal: Universidade de Trás-os-Montes e Alto Douro.

Disponível online em:
http://hdl.handle.net/10348/5344

https://www.researchgate.net/publication/259668770_Framework_for_Computer_Programming_in_Preschool_and_Kindergarten

2.4 A programação de computadores na educação

2.4.1 Comentários introdutórios sobre computadores e educação

Visto que esta tese se centra na educação pré-escolar, não apresentarei informações acerca da utilização de computadores ao nível do ensino secundário, superior ou em contextos de formação profissional (embora seja provável que muitas observações sejam válidas para todas as faixas etárias). Em vez disso, esta secção centra-se na utilização dos computadores no ensino de crianças, ou seja, níveis pré-escolar, primário/elementar/1.º ciclo do ensino básico, conforme as terminologias dos vários países. Nestes níveis concentra-se o groso da investigação de fundo e dos relatos de campo publicamente disponíveis, constituindo o panorama científico histórico geral. Não se discrimina aqui a informação relativa aos vários níveis, pois a utilização específica de computadores e da sua programação ao nível do pré-escolar (crianças entre os 3 e os 5 anos) é apresentada detalhadamente no capítulo 4.2, "Computers in preschool".

Dito isto, gostaria de assinalar que quando se menciona o uso de computadores – e da programação de computadores – na educação, a maior parte das pessoas que ouve esta expressão pensa acerca de novas matérias para ensinar e aprender, relacionados com os computadores. Ou então, pensa acerca de ferramentas de software ou equipamentos informáticos para apoiar o ensino de matérias tradicionais (ou em ambas as coisas). Ou seja, geralmente pensa-se em como as crianças podem beneficiar desta tecnologia e utilizá-la.

Contudo, esta tecnologia tem um impacte muito maior do que apenas na vivência das crianças: para atingir o seu impacte potencial, tem de se considerer não apenas o ponto de vista das crianças, mas também o dos professores. Os usos mais poderosos dos computadores na educação implicam mudanças, não apenas da parte dos alunos, mas também para os professores.

> *«* (…) *as questões* [teóricas subjacentes] *não se ficam apenas por um domínio, antes abrangem as preocupações tradicionais de filósofos, psicólogos, educadores e informáticos. De facto, o novo educador informático emerge como alguém com uma base em cada uma destas disciplinas. »*
>
> *(Solomon 1986, p. 1)*

2.4.2 A trama do uso de computadores na educação

O título desta secção tem um duplo sentido intencional. Apresento aqui um panorama geral de como os computadores têm sido usados – e ainda são usados – em ambientes de educação. Contudo, deve ser Claro para o leitor desta tese que me posiciono, pessoalmente, ao lado dos defensores do imenso potencial do uso de computadores para o desenvolvimento da educação. Para que este potencial seja concretizado, os computadores têm de ser usados de formas que não são generalizadas na atualidade, caso contrário o seu impacte torna-se negligenciável. Entre os recursos sobre este tema, aponto as obras de Papert (1998), Jonassen (2000) e diSessa (2000, cap. 9, "Stepping Back, Looking Forward").

Vários autores propuseram diferentes classificações para o uso dos computadores na educação (Berg, 2003, p. 14; Hoić-Božić, 1997, cap. 3.1.1; Jonassen, 2000, pp. 3-9; Solomon, 1986, pp. 8-13; Solomon, 1996, pp. 6-8). A visão abrangente que aqui apresento agrupa a utilização educative dos computadores em quatro grandes modelos, que serão explicados nas secções que se seguem:

- Aprender acerca dos computadores
- Aprender dos computadores
- Aprender com os computadores
- Aprender o pensamento, com computadores

O primeiro modelo, **aprender acerca dos computadores**, é a mera inclusão do computador e das suas características (equipamento, software) como novas temáticas para ensino e estudo. Embora conhecer uma ferramenta seja útil para compreender como ela pode ser usada de forma mais eficientes, esta abordagem visa apenas fazer das crianças melhores utilizadores de computadores; não está a ocorrer qualquer alteração nos métodos de aprendizagem. Neste modelo, a programação de computadores é aprendida pelos seus próprios méritos, visto fazer parte do corpo de conhecimentos da informática.

O segundo modelo, **aprender dos computadores**, representa o tipo de utilização educativa da informática que ocorre à maior parte das pessoas quando se usa a expressão "software educativo", tendo também sido definido como sendo o uso de computadores como "*livros escolares interativos*" (Solomon, 1986, pp. 8-10; Solomon, 1996, pp. 6-8). Este modelo representa o uso de aplicações de software ricas em informações (CD-ROM, sítios Web, etc.) e outras aplicações multimédia baseadas em informações. Também inclui a instrução apoiada por computador ("Computer-Aided Instruction", CAI), a aprendizagem apoiada por computador ("Computer-Aided Learning", CAL) e a formação baseada em computadores (Computer-Based Training, CBT), que utilizam percursos de aprendizagem e aplicações baseadas em treino e exercícios, iniciações ou "tutoriais" e os Sistemas de tutoria inteligente (Hoić-Božić, 1997, cap. 3.1.1; Jonassen, 2000, pp. 4-7). Neste modelo, a utilização de programação de computadores é escassa (se sequer existir).

O terceiro modelo, **aprender com computadores**, baseia-se na utilização dos computadores como ferramentas de aprendizagem, tal como o faríamos com uma folha de papel, um lapis, tintas, folhas secas, um microfone e um gravador, ou qualquer outra coisa. Está associado à transição moderna do ensino instrutivista (onde o professor é a fonte que fornece todo o conhecimento) para o ensino construtivista (onde o professor age como parceiro e coordenador, suportando a construção e desenvolvimento mentais do conhecimento por parte dos alunos). Outra forma de o dizer é que o computador deixa de ser um professor e começa a ser um parceiro de aprendizagem (Jonassen, 2000, p. 7). Este modelo também inclui o uso de simulações e jogos ricos em conteúdo, que as crianças podem explorer, sendo-lhes assim introduzido novos temas de forma interessante (Hoić-Božić, 1997, cap. 3.1.1). Neste modelo, a programação é uma das várias formas de utilização dos computadores.

O quarto modelo, **aprender o pensamento, com computadores** é semelhante ao terceiro, mas tem como preocupação central ajudar os aprendizes a melhorar os seus processos de aprendizagem. O nom epode ser enganador, pois parece menos genérico que os anteriores modelos: lida com "aprender o pensamento" …com computadores. "Então", poderia alguém perguntar, "este modelo só serve para aprender acerca do pensamento?" Sem dúvida, é isso mesmo. Mas como Papert o disse:

> *« Não se consegue pensar acerca do pensamento, sem pensar no pensamento acerca de algo. »*
>
> *(Seymour Papert in Minsky, 1988, p. 6)*

Por outras palavras, se alguém aprender acerca do pensamento, tal faz-se raciocinando acerca do seu próprio pensamento acerca de temas específicos. Esses temas não foram estudados só por acaso: no processo de estudo, o aprendiz ganha perceções acerca dos seus próprios processos mentais, que ocorrem enquanto trabalha nessa área. Isto, por sua vez, ajuda a criar ligações ponderosas entre

as abordagens pessoais acerca daquilo que se está a estudar. A programação de computadores é central a este quarto modelo, pois permite às crianças ensinar o computador. E ao fazê-lo, para encontrar e corrigir equívocos nos seus ensinamentos ("bugs"), têm de ponderar os próprios processos de pensar.

> *« Aprender com os bugs e debater estratégias de ensino para melhorar o que o computador sabe ajudam as crianças a refletir sobre a sua própria aprendizagem. »*
>
> *(Solomon, 1996)*

2.4.3 Aprender dos computadores

Este modelo de uso dos computadores na educação é o mais antigo, tanto em termos de aplicação prática como de bases teóricas. É acima de tudo uma extensão aos métodos educativos tradicionais, não uma mudança radical. Por este motivo, esta secção também apresenta uma breve história do uso dos computadores para fins educativos. Embora possa atualmente parecer limitado, este modelo trouxe de facto inovações importantes desde o primeiro momento, sendo ainda o mais utilizado na atualidade.

A lógica geral deste modelo pode ser encontrada na forma como Edward Thorndike, em 1912, apresentou as suas ideas de uso de uma máquina para melhorar a instrução:

> *« Se, por um milagre do engenho mecânico, um livro pudesse ser preparado para que apenas àquele que tivesse feito o que se dizia na página um ficasse visível a página dois, e assim sucessivamente, muito daquilo que agora requer instrução personalizada poderia ser alcançável pela folha impressa. »*
>
> *(Thorndike, 1912, p. 165, in McNeil, s.d.)*

Figura 37 – Edward L. Thorndike
1874 – 1949
De: http://www.psychology.uiowa.edu/Faculty/wasserman/Glossary/skinner.jpg

Thorndike reconhecia as vantagens do ensino baseado no professor, que para ele eram acima de tudo a maior empatia e personalização possibilitadas pelo contacto humano: ele considerava as palavras ditas mais cativantes do que "*marcas pretas visíveis*" (Thorndike, 1912, p. 161, in McNeil, s.d.), ou seja, do que a educação baseada em livros. Ele também encarava a ação do professor como uma vantagem da educação baseada em professores sobre a educação baseada em livros: atividades de treino, perguntas, explicações, indicações, etc. Mas aplaudia as vantagens do estudo baseado em livros, pois este "*permite a um estudante pensar ao seu próprio ritmo, obter os factos uma e outra vez, as vezes que precisar, testar-se ponto a ponto à medida que avança e fazer apontamentos*" (Thorndike, 1912, p. 161, in McNeil, s.d.). Contudo, ele via limitações no uso dos livros, que soam extremamente válidas ainda hoje:

> *«* [Os livros escolares] *geralmente fornecem os resultados do raciocínio e talvez problemas que exigem raciocínio, mas não conseguem gerir este último para que o aluno seja ajudado, em casa etapa, apenas o estritamente necessário para lhe permitir ajudar-se a si próprio, tanto quanto seja economicamente possível... nem costumam dar trabalho de dedução, organizado para estimular o aluno, para que faça e teste as suas próprias inferências. »*
>
> *(Thorndike, 1912, pp. 164-165, in McNeil, s.d.)*

Thorndike propôs que houvesse livros que incluíssem dado, instruções sobre a realização de experiências e trabalhar problemas sobre os mesmos dados, e ainda perguntas sobre inferências que

pudessem ser extraídas. O problema, Segundo ele, era que os alunos não seguiam as instruções escritas e tentavam ler tanto os problemas como as respostas, em vez de tentarem desenvolver as suas próprias respostas. Foi para ultrapassar isto que ele propôs a criação de uma máquina que controlasse o ritmo de estudo, como na citação fornecida anteriormente, no início desta secção.

Figura 38 – Pressey Teaching Machines
De: http://www3.uakron.edu/ahap/apparatus/images/photos/presseyteachingmachines(8-8-2).jpg

A primeira tentativa de pôr em prática as ideias de Thorndike ocorreu dez anos depois, no início dos anos 1920. Um professor universitário de psicologia educativa, Sidney Pressey (1888 – 1979), desenvolveu uma máquina de testes (*vd.* Figura), para ajudar os alunos a treinar e avaliar os seus conhecimentos (McNeil, n.d.; McDonald *et al.*, 2005). Ele queria que os professores se centrassem em "*atividades inspiradoras e estimuladoras do pensamento, que são, presume-se a verdadeira função do professor*" (Pressey, 1926, p. 374, in McNeil, s.d.). A sua máquina "*parecia um carreto de máquina de escrever com uma janela que revelava uma pergunta"* (McNeil, s.d.). Para responder, "(…) *o aluno consulta um item numerado, num teste de escolha múltipla. Ele ou ela carrega no botão que corresponde à primeira escolha de resposta. Se for a correta, o aparelho avança para o próximo item; se estiver errada, o erro é contabilizado e o aluno tem de continuar a fazer escolhas até chegar à resposta certa*" (Skinner, 1968, cap. 3). Pressey continuou a desenvolver as suas máquinas de ensinar até 1932, altura em que a Grande Depressão levou ao congelamento do desenvolvimento de tecnologias educativas, só retomado após a Segunda Guerra Mundial (McDonald *et al.*, 2005).

Cont8udo, apesar do desenvolvimento de tecnologias educativas ter ficado congelado neste período, foi uma era em que se desenvolveram ideias cruciais de disseminação da informação, que mais tarde integrariam as tecnologias e práticas educativas modernas. Estas ideias só muito mais tarde se concretizaram, quando surgiram os computadores pessoais e a Internet. Uma dessas ideias foi o conceito de H. G. Wells (1937) de uma enciclopédia mundial, permanentemente atualizada, em contacto com todas as instituições relacionadas com o saber:

> *« Tanto a recolha como a distribuição de saber pelo mundo são atualmente extremamente ineficazes; e os pensadores* (…) *começam a compreender que pelo contrário, a linha mais esperançosa de desenvolvimento da nossa inteligência racial resida no sentido de criar um novo organismo mundial para a recolha, indexação, resumo e disponibilização do conhecimento* (…). *Estes inovadores* (…) *projetam um organismo mundial unificado, se não mesmo centralizado, para "agregar a mente do mundo", que não será tanto um rival das universidades, mas um suplmeneto, um acrescento Coordenador das suas atividades educativas – à escala planetária.* (…) *Toda a memória humana pode ser – e provavelmente será, num curto espaço de tempo – disponibilizada a cada indivíduo.* (…) *Não tem de ser concentrada em nenhum lugar específico. Não tem ser vulnerável, como o são uma cabeça ou um coração humanos. Pode ser reproduzida completamente e com exatidão, no Peru, na China, na Islândia, na África Central ou em qualquer outro local que pareça proporcionar uma garantia contra o perigo e a interrupção. »*

Outra ideia foi o conceito de Vannevar Bush de uma máquina e do seu funcionamento, que ajudaria os seres humanos a navegar mais facilmente pelas vastas quantidades de informação produzidas no mundo. Ao fazê-lo, ele anteviu o uso dos computadores como sistemas pessoais para gestão da informação (isto numa altura em que a sua única finalidade era fazer cálculos):

> *« Considerem um aparelho futuro, de uso individual, semelhante a uma biblioteca ou arquivador privados. Precisa de um nome; e para cunhar um ao calha, "memex" há de servir. Um memex é um aparelho onde um indivíduo armazena todos os seus livros, registos e comunicações; um aparelho que é mecanizado, para que possa ser consultado com velocidade e flexibilidade excecionais. É um suplemento íntimo e aumentado da sua memória.*
>
> *Consiste numa secretária. E embora possa presumivelmente utilizado à distância, é em primeiro lugar a peça de mobiliário em que trabalha. Sobre ela estão ecrãs inclinado, translúcidos, nos quais se pode projetar materais, para leitura cómoda. Há um teclado e conjuntos de botões e alavancas. Mas de resto, parece-se com uma vulgar secretária. »*
>
> *(Bush, 1945, p. 10)*

Outro importante contributo de Vannevar Bush no mesmo artigo está estreitamente relacionado com a educação: ele apercebeu-se de que o raciocínio baseado numa coleção pessoal de factos memorizados já não era suficiente para lidar com uma sociedade onde a quantidade de informação era vasta. Uma consequência não explícita deste ponto de vista, em termos educativos, é que ser capaz de procurar, analisar e julgar as informações já não era apenas meramente útil, mas antes tornara-se uma aptidão absolutamente crucial no mundo moderno.

> *Presume-se que o espírito humano deveria elevar-se se melhor puder rever o seu passado sombrio e analisar em plenitude e objetivamente os seus problemas presentes. Ele construiu uma civilização tão complexa que precisa de mecanizar os seus registos de forma mais complete, caso deseje levar a sua experiência até à sua conclusão lógica, não apenas ficar atolado a meio caminho, sobrecarregando a sua memória limitada. A sua excursão pode ser mais agradável se puder readquirir o privilégio de esquecer a miríade de coisas que não precisa de ter imediatamente à mão, com alguma garantia de que pode voltar a encontrá-las caso se revelem importantes.*
>
> *(Bush, 1945, p. 13)*

Figura 39 – Burrhus Frederic Skinner, 1904 – 1990
De: http://www.psychology.uiowa.edu/Faculty/wasserman/Glossary/skinner.jpg

Após a Segunda Guerra Mundial, os computadores eletrónicos tinham já surgido (cf. secção **Erro! A origem da referência não foi encontrada.**) e logo foram adaptados para fins educativos. As primeiras tentativas de utilizer os computadores para apoiar a aprendizagem estavam fortemente ligadas às teorias comportamentalistas, particularmente a Teoria do Condicionamento Operante de B. F. Skinner (Kearsley, 2004). Esta teoria centra-se nas consequências das ações específicas de cada pessoa. Em vez de analisar a resposta decorrente de um estímulo específico, Skinner concentrou-se nas consequências dos atos. Segundo a sua filha, enquanto trabalhava com ratazanas, ele "*descobriu que o ritmo com que a ratazana premia a barra* [junto dela] *não dependia dos estímulos que o precediam* (…) *mas no que se seguia à pressões sobre a barra.* (…) *este tipo de comportamento atuava no ambiente e era controlado pelos seus efeitos. Skinner chamou-lhe comportamento operante. Ao processo de organizar as contingências do reforço responsável pela produção deste novo tipo de comportamento chamou condicionamento operante.*" (Vargas, n.d.).

Depois de desenvolver o seu saber sobre o condicionamento do comportamento animal, ele apercebeu-se de que as ações dos professores humanos eram ativamente contrárias aos princípios do condicionamento operante: a quantidade de informação apresentada aos alunos iria exigir grandes alterações comportamentais de uma só vez, e a reação proporcionada aos alunos (avaliação das

respostas em testes) ocorria muito depois da apresentação das informações (McDonald *et al.*, 2005). A partir destas observações, ele decidiu que o condicionamento operante podia ser usado para melhorar o ensino e a aprendizagem.

Figura 40 – Máquina de ensinar de Skinner
De: http://www.americanhistory.si.edu/teachingmath/images/1999-01829.jpg

Skinner apresentou as suas ideias para melhorar a aprendizagem humana num artigo de 1954 com o título "A ciência da aprendizagem e a arte de ensinar" (Reiser, 2001, p. 59). Para usar a teoria comportamentalista na instrução, o comportamento de cada criança tinha de ser moldado de acordo com cada um dos atos de aprendizagem da criança. O professor tinha de preparer a sequência correta de pequenos pedaços de informação, apresentados como problemas, para que a sua correta resolução pudesse recompensar a criança com um estímulo positivo[1].

Mas numa sala de aula normal, com 20 a 30 alunos, pouca hipótese há de o professor fazer isto. Por isso, Skinner concebeu uma máquina de ensinar que apresentaria problemas aos alunos para que os resolvessem (Figura). Tais máquinas, a primeira das quais foi apresentada em março de 1954 numa palestra na Universidade de Pittsburgh (NMAH-BC, 2002), forneceriam as informações em pequenos pedaços e a reação era produzida de imediato, após cada resposta, para permitir o condicionamento do aluno. Embora esta primeira máquina numa tenha sido usada numa sala de aula, outras foram desenvolvidas, tanto mecânicas como informáticas, tendo o seu uso acabado por resultar no que veio a designer-se o **movimento da instrução programada** (Vargas, s.d.). Uma destas primeiras máquinas baseava-se num computador: o IBM 650 (Figura e Figura), "*o primeiro computador empresarial comercializado pela IBM*" (Cruz, 2004).

Figura 41 – IBM 838 Inquiry Station para o IBM 650
De: http://www-1.ibm.com/ibm/history/exhibits/650/images/3403ph16.jpg

> *« O IBM 650, um computador digital de alta velocidade, que tinha uma máquina de escrever como interface, era usado como máquina de ensinar. »*
>
> *(McNeil, s.d.)*

Devido ao enorme preço do IBM 650, que for a desenvolvido para utilização empresarial e científica, o seu uso educativo obviamente não foi disseminado. Mas assinalou o início da utilização educativa dos computadores.

Figura 42 – Unidade de consola IBM 650
De: http://www-1.ibm.com/ibm/history/exhibits/650/image

Estas origem comportamentalistas originaram uma longa linha de desenvolvimento de programas informáticos para a educação, que partilharam a mesma base filosófica. Software deste género é simples de programar e desenvolver (Jonassen, 2000, p. 5) e mesmo hoje muitos materiais educativos para computadores pessoais empregam de alguma forma este princípios.

Historicamente, uma figura incontornável envolvida no desenvolvimento deste tipo de software foi Patrick Suppes. Em 1963, ele e os seus colegas do Instituto para Estudos Matemáticos nas Ciências Sociais (IMSSS), da Universidade de Stanford, iniciaram um projecto de investigação acerca do uso de computadores na educação, seguindo orientações comportamentalistas. Quatro anos depois, em 1967, fundaram a Computer Curriculum

[1] Está disponível online um programa de exemplo de autoinsturção, segundo este método, para uso em sala de aula por instrutores. Pode ser usado em computadores pessoais com o sistema Microsoft Windows, a partir do sítio Web da Fundação B. F. Skinner, neste endereço: http://www.bfskinner.org/instruction/setup.exe.

Corporation (CCC)[2], orientada para a comercialização e distribuição de materiais educativos desenvolvidos nos anos precedents, bem como para criação de novos materiais (Solomon, 1986, pp. 16-30; Solomon, 1996, p. 31).

Figura 43 – Patrick Suppes 1922 –
De: http://www.booknoise.net/flickeringmind/characters/src/patrick.jpg

No início e meados dos anos 1960, a utilização daquilo a que agora se chama multimédia, como gráficos, som, vídeo, etc. não estava disponível num ecrã de computador ou era extremamente dispendioso. Por este motive, os primeiros materiais só recorriam a texto. Uma interação habitual com um aluno consistiria numa pergunta, como a que se segue (extraída de Solomon, 1996, p. 32):

7 x 6 = _____

Uma resposta correta apresentaria ao aluno parabens e um novo exercício. Uma resposta errada proporcionaria uma reação diferente, que nos materiais originais da CCC seria algo como:

TENTE OUTRA VEZ

O aluno seria então confrontado com o mesmo exercício. Outra resposta errada resultaria na apresentação do resultado e da mensagem "*TENTE OUTRA VEZ*". Se a terceira tentativa também estivesse errada, o aprendiz passaria para um exercício diferente, provavelmente com um nível de exigência mais baixo.

Embora os ambientes gráficos e sonoros tenham tido enorme evolução desde essa época do final dos anos 1950 e início do anos 1960, a filosofia básica mantém-se igual: divider o conhecimento em pequenos pedacinhos, apresentá-los numa dada sequência e apresentar perguntas ao aluno. Se o aluno acertar, ele/ela avançar para outros pedacinhos, possivelmente mais complexos. Se o aluno errar, ele/ela terá de rever ou estuary pedacinhos anteriores de conhecimento e responder a perguntas sobre eles, até que o sistema avance. As versões mais recentes do software que usa esta filosofis podem apresentar um cenário visual e pedir ao aluno que escolha um objeto em falta ou que juntem notas para pagar uma compra, em vez de lhes pedir para multiplicarem 7 x 6, com parabens sonorous e indicações visuais. (Há exemplos com capturas de ecrã em Solomon, 1996, pp. 33-35 e 40-41.)

Esta categoria de aplicações, conhecida como "**instrução e treino**", or "**ensino apoiado por computador**" (*Computer-Aided Instruction*, CAI) pode produzir resultados mensuráveis, pois dá-se bem a testes formais – de facto, os testes são cruciais à sua conceção. Assim, o impacte nos resultados dos alunos em exames formais, escritos, pode ser avaliado. Desde o início, uma ideia central no uso de aplicações de CAI em ambientes educativos foi a existência de um sistema de gestão para o educador/professor rever a evolução do aluno, que respostas ele/ela acertou ou errou e assim por diante. Essas informações permitiriam ao professor focar esforços relativos a um aluno, face às áreas de conteúdo onde o aluno demonstra menos facilidade de evolução. (Como exemplo, os princípios do sistema de gestão das aplicações da CCC são analisados por Solomon, 1986, pp- 16-30.)

Foi demonstrado que o uso de CAI melhora, no mínimo, os resultados nos testes dos alunos com piores notas:

> *« O currículo da CCC tem o seu êxito mais claro junto dos alunos com desempenho inferior ao seu ano escolar, de acordo com os resultados que têm em testes padronizados.* (…) *Estudos de Avaliação conduzidos entre 1971 e 1977 apontam para um padrão* (…) *os alunos apresentam uma melhoria na*

[2] A CCC foi comprada pela Simon and Schuster Publishing, que depois se tornou parte da Viacom, tendo então sido vendida pela Viacom à Pearson plc (Viacom, 1998). Atualmente (julho de 2004), o seu saber-fazer integra a divisão da Pearson designada Pearson Digital Learning (http://www.pearsondigital.com).

> *posição em temos de anos escolares de 1 a 1.5 no final da sua experiência em matemática, e cerca de 0,6 anos em leitura ou prática linguística*[3]*. »*
>
> *(Solomon, 1986, p. 25)*

Há, contudo, alguma controvérsia relativamente ao seu impacte além do conjunto específico de aptidões tidas em conta no conteúdo de ensino, bem como na sua transferência ou uso em novas áreas de conhecimento:

> *« Em 1982, uma Avaliação do Serviço de Testes Educativos*[4] (…) *concluiu que os estudantes obtinham ganhos significativos nas suas competências informáticas. As matérias de leitura e prática linguística não refletiam o mesmo tipo de maior melhoria nos resultados de testes. O resultado dos testes aumentou, de facto, no primeiro ano; mas nos anos seguintes não houve aumento (nem diminuição) dos resultados nos testes. »*
>
> *(Solomon, 1986, p. 25)*

As críticas a este método de educação baseada em computador costumam basear-se no facto de que "*os princípios comportamentalistas subjacentes à instrução e treino não conseguirem ter em conta, quanto mais alimentar, o pensamento complexo necessário para a aprendizagem significativa que permite resolver problemas, transferir competências para novas situações, construir ideias novas e originais, etc*." (Jonassen, 2000, p. 5). Os seus defensores argumentam que a instrução e o treino alimentam o automatismos; e que para que se possam aprender as competências de alto nível, as de baixo nível têm de estar automatizadas (Jonassen, 2000, p. 5; Solomon, 1986, p. 26). Creio que ambos os argumentos são verdadeiros: provavelmente será necessário algum nível de instrução e repetição para que os conhecimentos basilares fiquem tão entranhados que se possa avançar para os usar em tarefas mais complexas. Contudo, embora o treino faculte esse entranhamento do saber, é mais difícil antever como é que pode ajudar os alunos a transferir esse saber para novas situações e novos problemas.

A combinação de instrução e treino, contudo, não é o único método atual para aprender a partir dos computadores: a maior parte dos utilizadores de computadores certamente terá passado por pelo menos outro método comum, a que se costuma chamar "**tutorial**". Um tutorial clássico apresenta informações, coloca perguntas (geralmente com as respostas num sistema de escolha múltipla, para evitar as dificuldades de interpretação da linguagem escrita). Se um estudante acertar, ele/ela é recompensado (como na instrução e treino). As respostas erradas levam a alterações no percurso de instrução, para proporcionar uma alternativa, uma correção do erro, sendo mais tarde apresentada novamente a pergunta que foi anteriormente mal respondida (Jonassen, 2000, pp. 5-6).

Um tipo específico de tutorial que se encontra atualmente é o que se destina a treinar um ser humano quanto ao uso de alguma aplicação de software específica, ou ao treino de alguma situação que pode ser modelada ou simulada num computador. Por exemplo, muitos videojogos são significativamente complexos e incluem tutoriais para apoiar o utilizador a aprender a jogá-los. Estes tutoriais costuam adotar uma abordagem prática, em vez de uma abordagem de perguntas e respostas, mas mantém-se a filosofia de avançar um passo de cada vez: o programa pede ao utilizador que faça uma tarefa específica, como introduzir texto numa caixa de edição específica, clicar num botão ou arrastar um ícone; e geralmente evita as opções "erradas". Se o utilizador tentar fazer algo diferente do que se pretende, o programa tutorial costuma alertar quanto à escolha errada e conduzir o utilizador para o percurso adequado, por vezes com explicações adicionais. Avançando ao longo do tutorial, o utilizador pode aprender a distribuir tropas, gerir recursos, desenvolver uma civilização, sustentar um

---

[3] Nota de tradução: no original, *language arts*, consistindo em aspetos como o debate, a escrita, a interpretação da oralidade, entre outros.

[4] Nota de tradução: *Educational Testing Service*, uma organização privada norte-americana sem fins lucrativos dedicada a testes e avaliações educativas. https://www.ets.org/

grupo de pessoas e a negociar de forma ética com mercadorias intergaláticas ou qualquer outro tipo de mecânica de jogo que seja necessária.

Este nível de interação ressoa assinalavelmente àquela afirmação de Thorndike citada no início desta secção, onde ele desejava que "*apenas àquele que tivesse feito o que se dizia na página um ficasse visível a página dois*" (Thorndike, 1912, p. 165, in McNeil, s.d.). É de facto uma técnica ponderosa, assegurar que o aprendiz de facto efetua todas as ações pretendidas e treina todos os pedacinhos de conhecimento considerados cruciais. A sua maior limitação é que não consegue acomodar diferentes necessidades de aprendizagem: caso algum aprendiz queira explorer uma área diferente do conhecimento a meio do percurso do tutorial, caso algum conceito específico não seja percebido pelo aprendiz, não há a opção de explorer mais ou testar o conceito: a única solução que e apresenta ao aprendiz é avançar ao longo do caminho previamente estabelecido. Por outras palavras, os tutoriais restringem a forma pela qual é suposto que os aprendizes construam os seus próprios significados: eles têm de acomodar a interpretação que outra pessoa fez do mundo e dos factos.

> *« Os alunos não são encorajados, nem sequer conseguem, determinar o que é importante, refletir ou avaliar o que sabem, nem construir quaisquer significados pessoais para o que estudam. O que adquirem dos tutoriais, demasiadas vezes, é conhecimento inerte, porque não o estão a aplicar. »*
>
> *(Jonassen, 2000, p. 6)*

Contudo, os videojogos modernos continuaram a desenvolver o conceito de tutorial, sendo que atualmente o utilizador pode ter maior grau de liberdade quanto à sua direção de aprendizagem e de atuação. Presentemente, não é incomum que o tutorial de um jogo ocorra durante o próprio ato de o jogar, com o sistema tutorial a fornecer indicações e sugestões, conforme a evolução do jogo. As imagens que se seguem são ecrãs do tutorial do jogo Civilization III (Figura), da Infogrames (Firaxis, s. d.). O tutorial deste jogo é um belo exemplo de como um jogador pode beneficiar de pedacinhos de aconselhamento e indicações enquanto joga verdadeiramente um jogo.

Figura 44 – Caixas do Civilization III

De: http://www.firaxis.com/images/interface3.0/gamespage_box_civ3.jpg

Figura 45 – Ecrã de entrada do Civilization III

De: http://www.firaxis.com/images/interface3.0/gamespage_box_civ3.jpg

A Figura 45 apresenta a opção do tutorial realçada entre as opções do ecrã de entrada no jogo. Após escolhê-la, o jogador inicia o jogo normalmente, sendo-lhe apresentado apenas um pedacinho de informação acerca da situação inicial e dos objetivos (Figura 46).

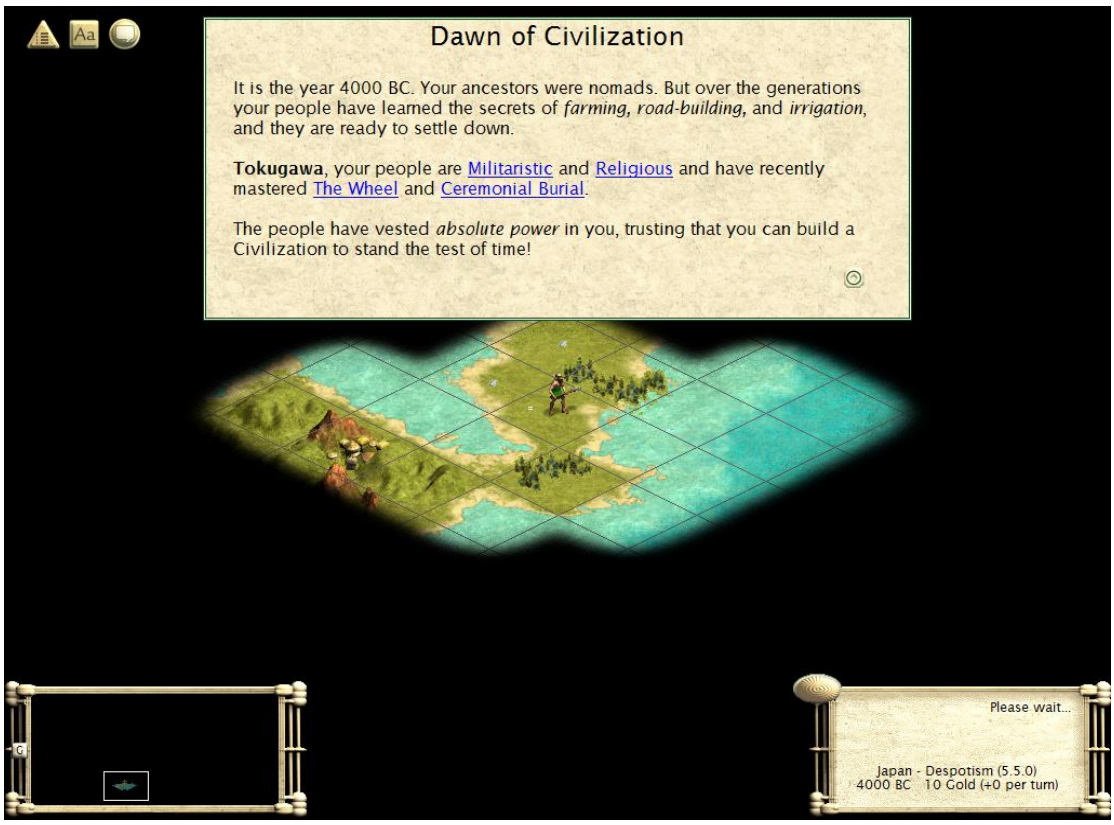

Figura 46 – O primeiro pedacinho de informação do Civilization III: situação inicial e objetivos

Em breve, contudo, chegam as primeiras indicações: o jogador tem de mover a primeira unidade (um "colono") em direção a um bom local para fundar a sua capital. O jogo também dá conselhos acerca do que constitui "um bom local" e sobre os comandos para mover o colono (Figura 47).

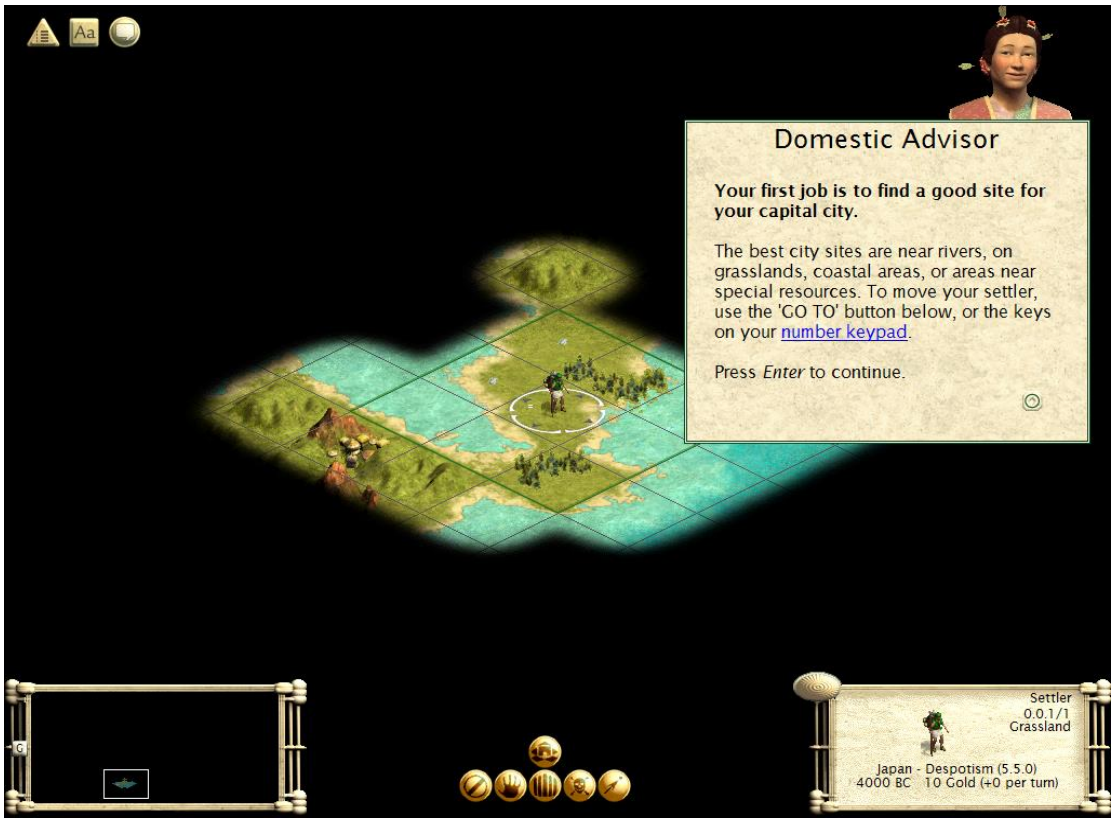

Figura 47 – As primeiras indicações do Civilization III: encontrar um bom local para fundar a capital

À medida que o jogo evolui, o sistema do tutorial destaca elementos no ecrã que o utilizador tem de usar nessa altura e fornece-lhe explicações (Figura 48).

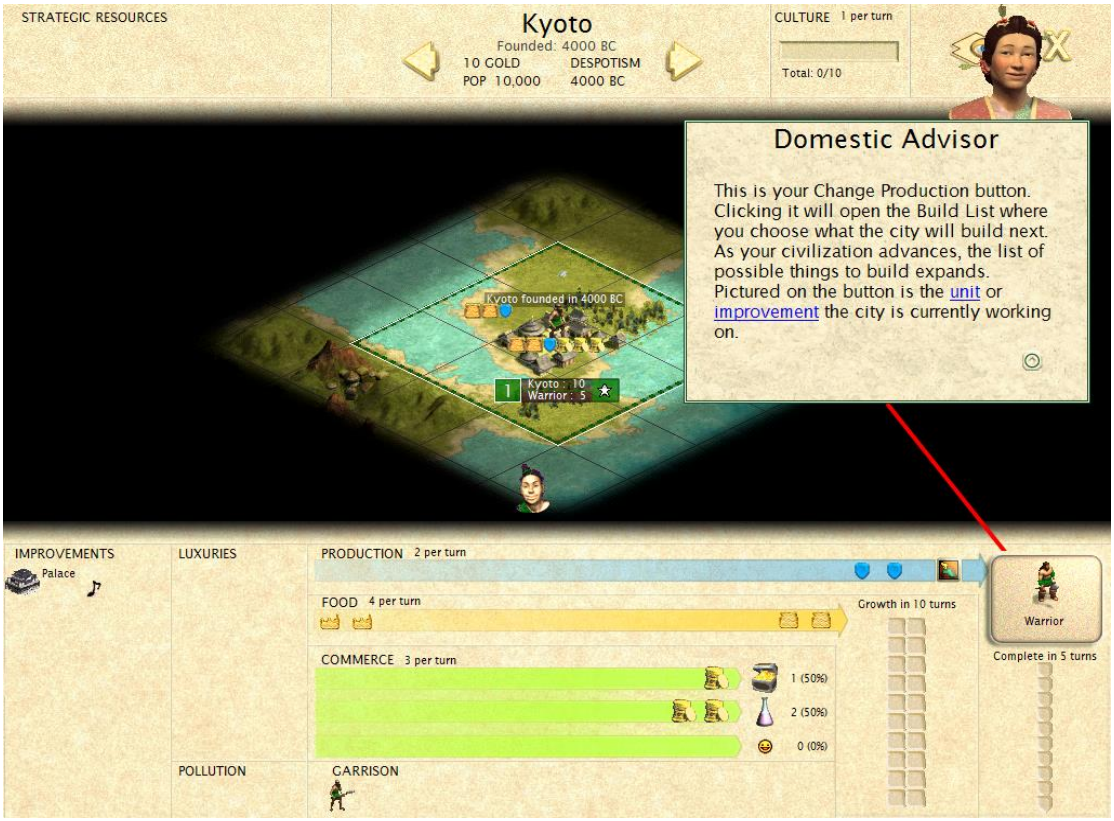

Figura 48 – A explicação do tutorial do Civilization III sobre os elementos da interface

Em todas estas figuras, o texto do tutorial contém elementos a azul, sublinhados, que representam hiperligações com destino a mais informações, com maior extensão. A Figura 49 apresenta um exemplo de um menu de informações atingido através da hiperligação *improvement* (melhoria [da cidade]) da Figura 48. Pode-se clicar em cada elemento para obter informações específicas a ele.

Figura 49 – Exemplo de ecrã do Civilization III com informações hiperligadas

O utilizador pode jogar como o faria normalmente, sem restrições. À medida que o jogo avança, o sistema de tutorial apresenta-lhe informações aparentemente relacionadas com a situação atual e instruções ou recomendações quanto a ações que devem ser tomadas (Figura 50). Digo "aparentemente", porque a sequência delas raramente varia: começa-se por dizer ao jogador que tem de construir estradas em redor da cidade, depois que deve construir uma mina nalgum quadrado e por aí adiante. Cabe ao jogador decider se o faz ou não, mas o tutorial não reage a isso (uma forma mais

rígida e limitada de tutorial, comum a muitos programas, evitaria que o utilizador/jogador avançasse se não agisse segundo as instruções do tutorial).

Figura 50 – Sugestões e instruções do Civilization III com o jogo tutorial em pleno curso.

Se o utilizador não ler, não ponderar nem tiver em consideração o pedaço de informação fornecido pelo tutorial, não há consequência: o tutorial simplesmente fornecerá mais adianta o próximo pedacinho de informação. O jogador pode repetir o tutorial as vezes que quiser; e dada a natureza do jogo Civilization III, ser-lhe-ão apresentadas situações de jogo completamente novas. Contudo, a sequência de sugestões fornecida pelo tutorial será a mesma: como já se referiu, o objetivo do tutorial é vincular ao utilizador um conteúdo específico. O jogo depende da sua capacidade de aliciamento para que o jogador o fique a conhecer melhor e aprenda enquanto joga, possivelmente até regressando mais tarde ao tutorial ou as materiais de referência, para obter informações que façam sentido para a situação ou estratégia de jogo específica que está a tentar usar. Mas nessa altura, o tutorial não é a força motriz da aprendizagem: a sua relevência ficou exaurida nas fases iniciais, ao fornecer pedacinhos de conhecimento.

Há uma diferença clara nos estilos de aprendizagem, entre a fase de uso do tutorial e a fase de jogo. Em ambas, o jogador está a aprender a usar o jogo, mas sem dúvida é quando o tutorial já não está a ser usado que as perceções mais profundas estão a ser alcançadas. Contudo, é muito provável que tal nível de proficiência no jogo– e consequentemente, tais perceções –não viesse a ser atingido por muitos jogadores, se não fosse o apoio em estilo de "rodinhas de treino" fornecido pelo tutorial. E no entanto, se o conhecimento do jogo fosse analisado em testes escritos, é provável que o maior aumento em "conhecimentos" testados fosse registado durante o tutorial ou imediatamente após a sua conclusão. Isto é um exemplo claro do paradox e contínuo debate entre os apoiantes e críticos do estilo de educação "Aprender dos computadores". (E também, de forma crucial, de como ambas as partes têm argumentos válidos.) O estilo de aprendizagem que tem lugar durante o jogo propriamente ditto é exposto mais à frente, na secção seguinte: 2.4.4 – "Aprender com computadores".

As ideias combinadas do estilo CAI original e dos tutoriais conduziram a um desenvolvimento recente nesta abordagem à instrução baseada em computadores: os **sistemas de tutorial inteligente** (*intelligent tutoring systems*, ITS). Basicamente, estes sistemas adaptam os métodos de CAI ao aluno, mas também em termos do conteúdo, não apenas em termos do ritmo. Normalmente, estes Sistemas

empregam algum nível de técnicas de inteligência artificial para determinar as áreas de conteúdo onde o estudante pode estar a ter dificuldades, para lhe fornecerem sugestões personalizadas ou outros apoios textuais; alguns podem também criar, automaticamente, problemas e cenários específicos para cada aluno.

> *«* (…) *os ITS vão além das simulações de formação, respondendo às perguntas do utilizador e fornecendo indicações individualizadas. Ao contrário de outras tecnologias de formação apoiada em computador, os ITS avaliam as ações de cada aprendiz dentro destes ambientes interativos e desenvolvem um modelo do seu conhecimento, das suas aptidões e da sua competência. Com base no modelo de aprendiz, os ITS geram estratégias de instrução à medida, em termos tanto do conteúdo como do estilo, fornecendo explicações, indicações, exemplos, demonstrações e problemas práticos, conforme seja necessário. »*
>
> *(Ong & Ramachandran, 2003, p. 2)*
>
> *« Um tutor humano especializado ajusta continuamente o conteúdo e o modo de apresentação, com base na respostas recentes de um estudante e nos objetivos de ensino do tutor, bem como do conhecimento prévio do estudante. De forma semelhante, o software curricular apresenta ao estudante os objetivos em níveis escolhidos a partir de uma combinação de linhas curriculares, num modo selecionado, seja isso uma pergunta, um tutorial, uma breve apreciação ou outras formas de instrução. »*
>
> *(Thrall & Tingey, 2003, p. 1)*

Como aspeto final, refira-se que esta secção debateu pouco o papel do professor durante o processo educativo ou de aprendizagem, além das observações iniciais acerca da perspetiva de Thorndike (na página **Erro! Marcador não definido.**). E assim foi, porque este estilo de uso dos computadores na educação está concebido para ter lugar sem intervenção do professor, além da supervisão de criaças/estudantes. Considera que o papel do professor tem lugar antes e depois das atividades informáticas, não durante elas (antes, fornecendo conhecimentos ou preparando a sua apresentação; depois, avaliando a evolução do estudante). Os professores são vistos como tendo de ser libertados das atividades de instrução (que podem ser asseguradas pelos computadores) para desempenhar outras tarefas, mais significativas, com o objetivo geral de transtimir conhecimentos.

> *« Tal como a maior parte dos seus produtos da CCC, o programa não incluia muita interação com o professor. "Foi concebido para não precisar de um tutor", disse Suppes. "São apenas para resolução de problemas"— ou seja, para responderem, em geral por e-mail. Suppes* (…) [está] *simplesmente a tentar encontrar uma forma de contornar os professores fracos, que ele parece encarar com o calcanhar de Aquiles da educação. "Há muita treta sobre os professores. Não devemos pensar que são todos flores lindas prestes a desabrochar. Todos gostaríamos de ser tutorandos de Aristóteles. Mas tal não é possível." »*
>
> *(Oppenheimer, 2003)*

Assim, a lógica desta abordagem ao uso de computadores na educação inclui a noção de que os professores não precisam de estar envolvidos ativamente enquanto as crianças usam os computadores. Mesmo nos modernos sistemas ITS, esta perspetiva não se alterou: quando são desenvolvidos para uso em salas de aula, os ITS também apoiam o professor da sala, mas apenas por lhe fornecerem informações administrativas acerca da evolução dos estudantes. O principal papel do professor continua a ser antes ou depois da formação baseada em computador. Por exemplo, um recente relatório de investigação acerca do uso de um desses Sistemas apresentava o seu funcionamento da seguinte forma: o professor prepara o tutor informático, que depois toma conta do ensino; se o aluno

precisar de apoio, as opções do professor são substituir o tutor informático apenas para preparer o aluno para a ele regressar; ou manter-se quieto e deixar continuar o tutor informático.

> *« Para maximizer a aprendizagem, o tutor e o professor da sala têm de trabalhar juntos. Assim, o tutor, simulado pelo software curricular, deve adaptar-se aos objetivos do professor para a turma e para cada estudante individual, o que se consegue através do ajuste de estratégias do professor para o estudante, dentro de uma disciplina ou ao longo de várias. Além disto, o tutor deve partilhar com o professor informações acerca da evolução académica do estudante, o que se consegue através do sistema de relatórios do software curricular. Com base nestas informações, o professor pode decider quando quer dar ao estudante mais Liberdade ou mais estruturação; mais instrução direta ou mais exploração; mais foco ou mais variedade. Em alternative, o professor pode deixar o programa ajustar-se ao estudante e tomar as decisões acerca do que fazer a seguir. »*
>
> *(Thrall & Tingey, 2003, p. 1)*

Isto é coerente com o pano de fundo comportamentalista, que se preocupa acima de tudo com as causas observáveis e as reações, não com aquilo que para a teoria comportamentalista são os processos mentais internos inobserváveis, que por essa natureza são um debate meramente especulativo. Uma consequência desta perspetiva é que se supõe que os professores devem simplesmente preparar e transmitir o conhecimento, num ambiente e modo que siga a teoria comportamentalista, não devendo embrenhar-se em especulações acerca dos desenvolvimentos educativos que possam estar a ter lugar no interior da mente.

> *« Os professores são formados para instruir e não para educar. A formação profissional sobre instrução é possível, mas não sobre educação, dada a presente constituição da profissão docente e das instituições que os formam. »*
>
> *(Suppes, 1995, resumindo o pensamento sobre educação de Siegfried Bernfeld, com quem sente empatia)*

2.4.4 Aprender com computadores

Embora a abordagem à aprendizagem baseada em computadores descrita na secção anterior, "Aprender dos computadores", possua a história mais longa, cedo deixou de ser a única. Esta secção, "Aprender com computadores", apresenta um estilo de aprendizagem baseada em computadores que, essencialmente, vê os computadores como apoios à aprendizagem, que devem ser integrados no processo de ensino-aprendizagem e fornecer apoio à aprendizagem (e ao ensino), em vez de se tornarem a fonte do conhecimento e controlarm o processo de aprendizagem. Este diferente ponto de vista sobre o papel dos computadores na educação, em contraste com o ponto de vista comportamentalista anteriormente apresentado, está possivelmente associado com a tendência moderna para utilizar a expressão "Aprendizagem Aprimorada pela Tecnologia" (*Technology-Enhanced Learning*, TEL), em vez de referir ensino "baseado em computadores".

Figura 51 – Robert B. Davis, 1926-1997
De: http://www.rbdil.gse.rutgers.edu/bob_davis_01.jpg

As raizes desta abordagem ao computadores reside no trabalho de Robert B. Davis, professor de ensino da matemática na Universidade Rutgers, particularmente no projecto de reforma da matemática escolar que se iniciou em 1956, designado Projeto Madison. Este projecto "*está subjacente aos materiais para matemática na escola primária implementados pelo sistema*

informático [PLATO[5]] *da Universidade do Illinois no anos 1970*" (Solomon, 1996, pp. 7 & 43), sobre o qual falo mais à frente nesta secção.

Uma forma de deixar claras as diferenças entre estes estilos educativos, não é tanto apresentar o software (que, obviamente, costuma ser muito diferente – mas por vezes semelhante) mas antes discutir como se preconiza que seja usado. O objetivo de Skinner (e de Suppes) era substituir os professores; se não completamente, pelo menos durante o período de uso dos computadores, entregando a instrução aos computadores e libertando os professores para outras tarefas: era suposto que o computador fosse usado autonomamente. Davis, pelo contrário, vê o professor como um elemento ativo no processo educativo, um que tem de embarcar com as crianças num processo criativo, "*alicerçando-se sobre o que as crianças já sabem, ao mesmo tempo que se desafiam para inventor soluções para novas situações, através do uso inteligente e coerente de elementos manipuláveis*" (Solomon, 1996, p. 57). Isto, obviamente, também se aplica ao uso de computadores.

Figura 52 – Ecrã de uma aplicação educativa do PLATO, nos anos 1970
De: http://www.cineca.it/immagini/history/plato.jpg

Em resumo, não é tanto uma questão de descrever as ferramentas de software, mas antes de descrever como podem ser usadas como ferramentas educativas, e não como podem usar a criança. Vários conceitos introduzidos por Davis, neste contexto, influenciaram fortemente o desenvolvimento de software educativo, dois dos quais foram extensivamente usados pelo projecto Madison: a **estratégia de ensino paradigmático** e a **aprendizagem por descoberta**.

Segundo a estratégia de ensino paradigmático, "*as novas ideias são introduzidas através de exemplos cuidadosamente selecionados*" (Solomon, 1986, p. 32). Um professor procura ligar o conhecimento novo ao conhecimento pré-existente da criança, em termos práticos alicerçando-se nele. Um exemplo seria a introdução da noção de frações numéricas, ligando-a ao processo de partilha de uma barra de chocolate ou de rebuçados, ou seja, atividades com as quais as crianças já estão familiarizadas (Solomon, 1986, pp. 32 & 37-39). O software pode ser introduzido de forma semelhante, propondo atividades que façam ligações semelhantes. Por exemplo, a Figura apresenta um ecrã de um programa modern, descendente das ideias de Davis. Nele, as adições, divisões e frações são apresentadas no contexto de uma cozinha: a criança tem de combinar medidores para preparar adequadamente a receita.

[5] *Programmed Logic for Automated Teaching Operations*: "lógica programada para operações de ensino automatizado"

Figura 53 – Ecrã do software "The Quaddle Family", da PLATO Learning

Extraído do vídeo disponível em: http://www.plato.com/downloads/movies/QUADDLE_300.mov

Segundo a estratégia de aprendizagem por descoberta, "*um professor poderá chamar a atenção para um problema, mas a tarefa de inventor um método para lidar com o problema fica a cargo do estudante*" (Solomon, 1996, p. 44, citando Davis). Os professores apoiam tais estratégias de descoberta fornecendo pistas e ajuda. Esta abordagem visa reforçar o uso do raciocínio por parte das crianças, em lugar dos métodos enlatados, ajudando assim a criança a desenvolver a sua estrutura interna de conhecimentos. A interação do professor presta ajuda neste processo, tanto durante o desenvolvimento como durante a correção de pressupostos errados, quando as coisas não correm bem. Por exemplo, a Figura 54, na página 79, mostra um programa de software destinado a esta estratégia: as crianças exploram os conceitos dentro de um contexto específico (neste caso, uma exploração arqueológica onde é necessário descodificar palavras) e depois têm de resolver vários problemas, podendo experimentar várias estratégias.

O video publicitário da editor dá um resumo do conteúdo das atividades:

> *« Este programa complete sobre fonética proporciona a oportunidade de começar por atividades de sensibilização para os fonemas, envolvendo a combinação, segmentação, rima e identificação isolada de sons.*
>
> *Nas abrangente atividades de fonética, os estudantes participam em exercícios de descodificação, nos quais estudam palavras para identificar os sons consoantes iniciais, finais e intermédios, bem como os sons vocálicos longos e curtos.*
>
> *Nas atividades "família de palavras", os estudantes usam padrões de famílias de palavras para erguerem os seus próprios muros de palavras interativos, praticando depois a organização de palavras em famílias. »*
>
> *(Plato Learning Inc., 2004)*

Figura 54 – Ecrãs do software "The Three Decoders", da PLATO Learning

Extraído do vídeo disponível em: http://www.plato.com/downloads/movies/3_DECODERS_300.mov

Obviamente, o software inicial de Davis tinha o aspeto da Figura 52, não o aspeto modern das aplicações apresentadas na Figura 53 e na Figura 54, mas as principais ideias educativas apresentadas nos parágrafos anteriores já estavam desenvolvidas desde o início.

Como parênteses histórico, refira-se que o software inicial de Davis foi desenvolvido para um sistema informático designado PLATO (*Programmed Logic for Automated Teaching Operations*), cujo desenvolvimento foi iniciado em 1961 no campus da Universidade do Illinois em Urbana-Champaign, por Donald L. Bitzer e H. Gene Slottow. O objetivo original do PLATO era ser uma «'*máquina de ensinar' automática para ensinar os alunos, de vários níveis, a utilizer um computador muito rápido (...) o ILLIAC I, de cinco toneladas*" (Hutchinson, 2003). Através de várias gerações evolutivas, o desenvolvimento do sistema PLATO conduziu à criação de várias tecnologias revolucionárias, como os ecrãs de plasma para apresentação de gráficos e os ecrã sensíveis ao toque (Figura), que simplificaram a interação das crianças. O sistema PLATO era geralmente usado em ambientes de rede, o que levou a outras inovações no campo do software. Entre estas, incluem-se as mensagens instantâneas, o e-mail, os ícones emotivos (*emoticons*) e várias outras funcionalidades de colaboração (Dear, 2004). Um belo exemplo do seu impacte nos utilizadores e no ambiente geral da informática é este comentário feito ao jornal The New York Times por Ray Ozzie, criador do software Lotus Notes (que viria a ser o exemplo canónico de "teamware" ou software para equipas):

Figura 55 – Donald L. Bitzer
De: http://www.ncsu.edu/BulletinOnline/photos/bitzer.jpg

> *« Depois de acabar o curso, disse a mim mesmo, "Seja lá como for, vou construir software para recriar o ambiente interativo que usei com o Plato." Esse pensamento levou-me à criação do Lotus Notes, que está instalado em 100 milhões de computadores pessoais, bem como tudo mais que fiz. »*
>
> *(Ozzie, 2002)*

O sistema PLATO tornou-se um produto de êxito, comercializado pela empresa Control Data, que em 1976 adquiriu os direitos ao uso do nome PLATO. O material curricular do PLATO foi vendido à TRO Learning, Inc. em 1989, tendo esta empresa em 2000 mudado o seu nome para PLATO Learning, Inc., "*para tirar partido da força da marca PLATO no Mercado de software educativo*" (Plato Learning Inc. n.d.). A Universidade do Illinois continuous a desenvolver programas educativos baseados em computadores, mas como o nome PLATO já não estava disponível, o sistema acabou por ficar conhecido como NovaNET, em 1988 (Dozen, 1998). Ironicamente, a NovaNET é agora comercializada pela Pearson Digital Learning, o que significa que os expoentes desta secção e da anterior vieram a ficar debaixo do mesmo chapéu empresarial.

Figura 56 – Estação de trabalho PLATO IV com ecrã de plasma sensível ao toque
De: http://www.ece.uiuc.edu/ingenuity/503/images/PLATO_IVb.jpg

Desde os primeiros dias do PLATO, esta linha de utilização educativa da informátiva viu desenvolverem-se outras estratégias e novidades, sendo agora uma linha muito ativa.

Uma característica saliente, que se nota nos ecrãs das aplicações desta secção, é que parecem ser divertidos – ou seja, parecem ser jogos, não produtos educativos. É natural que assim seja: porque de facto são jogos. Atualmente, a maior parte do software educativo para crianças combina componentes educativa e de entertenimento, sendo isto uma consequência direta da estratégia de ensino paradigmático de Robert Davis, usando na educação as atividades quotidianas das crianças,

Como é evidente, grande parte das atividades quotidianas das crianças é brincar. Este tipo de software chama-se **edutenimento**, "*um lugar que apela às crianças para gostarem do que estão a aprender, com uma combinação de sons, animações, videos, textos e imagens*" (Druin, 1996, p. 64). É também um setor empresarial mundial multimilionário e em expansão[6].

Esta mistura de entertenimento e educação implica que este tipo de software seja usado dentro e fora da sala de aula, com ou sem apoio ou intervenção do professore. De facto, o seu primeiro alvo de comercialização são as famílias, não as escolas. Uma consequência desta necessidade de uso autónomo pelas crianças foi a evolução sustentada da conceção e simplificação das interfaces de utilização. Em 1996, ao descrever um produto chamado *My First Encyclopedia* ("A minha primeira enciclopédia") Allison Druin referia:

> *« Deita for a a interface tradicional de janelas sobrepostas, menus pendentes e botões, trocando-os por uma árvore.* (…) *há pouco texto com que as crianças do pré-escolar se debateriam: em vez disso, há videos de crianças que explicam as ideias e conduzem as atividades. »*
>
> *(Druin, 1996, p. 78)*

À luz destas mudanças, a versão moderna de aprender com computadores ocorre segundo duas perspetivas: **aprender com computadores autonomamente** e **aprender com computadores nas escolas** (ou nooutrs ambientes de aprendizagem com o apoio de professores). Embora esta distinção quanto ao uso também esteja presente no modelo anterior ("Aprender dos computadores") não era então tão relevante, porque nesse modelo o envolvimento do professor durante o uso do computador era nulo ou quase nulo.

Enquanto se aprende autonomamente, é crucial que o software informático possua **boas características de autoaprendizagem**. O aprendiz tem de ser capaz de perceber, por si só, como usar o software e tem de querer usá-lo tempo suficiente para que seja eficaz. Num ambiente com apoio do professor, estas características de autoaprendizagem do software não são tão relevantes, pois o professor ou um parceiro do estudente podem apresentar a aplicação de software ao aprendiz; e provavelmente prestar-lhe apoio e orientação durante o seu uso. Contudo, ser fácil de aprender é também uma característica útil para qualquer software destinado a utilização num ambiente educativo, caso contrário o aprendiz gasta mais tempo e esforço a aprender a usar o software do que o justificam os benefícios que possam advir dele (Jonassen, 2000, pp.18-19).

Há um motive inerente para usar a expressão "boas características de autoaprendizagem" quanto ao software para utilização autónoma; e "ser fácil de aprender" para software destinado a ambientes com apoio do professor. Nesta última situação, o software é acima de tudo uma ferramenta, que pode ser fácil, difícil ou quase impossível de compreender sem apoio adicional (seja esse apoio um professor, um manual ou um livro escolar); mas para aprendizagem autónoma, é imperioso que o software permita que seja aprendido, enquanto o utilizador está ativamente envolvido. O apoio adicional só é relevante para que o utilizador progrida mais depressa ou possa ir além de certas barreiras de complexidade, não para iniciar o uso do software. Os jogos são um bom exemplo: a maior parte das crianças começa a jogar um jogo de imediato, sem ler manuais ou instruções. Muitos jogos fornecem manuais ou tutoriais, mas quando as crianças (ou os adolescents ou adultos) pegam nelas, geralmente é já depois de estarem interessados no jogo propriamente dito.

> *"Um dia decidi que queria ajudar o meu filho a jogar Pajama Sam in No Need to Hide When It's Dark Outside*[7]*.* (…) *Decidi jogar eu mesmo o jogo, para que pudesse "orientar" o meu filho enquanto ele jogava. (Agora ele cobra-me um dólar de cada vez que tento "orientá-lo" enquanto joga um*

[6] "*As vendas totais de software e equipamento para Videojogos, nos Estados Unidos, alcançaram os 8,9 milhares de milhões de dólares, mais do que os 7,3 milhares de milhões de receitas de bilheteria do cinema; 6,6 milhares de milhões da receita dos videjogos adveio de vendas de software, em lojas de retalho e online*" (Poole, 2000, p. 6).

[7] "O Samuel Pijama não precisa de se esconder quando está escuro lá fora" (nota de tradução).

videojogo – ele diz que é "dar-lhe ordens" e "dizer-lhe o que há de fazer quando ele consegue descobrir sozinho.") »

(Gee, 2003, pp. 4-5)

James Paul Gee (2003, p. 6) chamava a isto "*ter bons princípios de aprendizagem integrados na sua conceção*" e os jogos (sejam de edutenimento ou não) são excelentes exemplos disto:

« Eis-nos então aqui com algo que é demorado, difícil e desafiante. Contudo, não se consegue jogar um jogo se não se conseguir aprendê-lo. Se ninguém jogar um jogo, ele não vende e a empresa que o faz vai à falência. É claro que quem os concebe podia fazer jogos cada vez mais curtos e simples, para facilitar a aprendizagem. Isso é geralmente o que fazem as escolas. Mas não: neste caso, os concetores de jogos fazem-nos cada vez mais demorados e desafiantes (...) *e ainda conseguem que possam ser aprendidos. »*

(Gee, 2003, p. 6)

Assim, em resultado tanto das experiências iniciais de adaptação da aprendizagem às atividades das crianças como do desenvolvimento de produtos no mercado dos videojogos, há hoje em dia esta classe de software de edutenimento e entretenimento que produz por si só uma experiência ativa de aprendizagem, para uma enorme variedade de conteúdo, tanto concreto ("conteúdo curricular", por assim dizer) como a um meta-nível (competências de aprendizagem, exploração e raciocínio)[8].

A outra abordagem, aprender com computadores nas escolas[9], evoluiu a partir da abordagem de Robert Davis à educação baseada em computadores, onde o professor assume um papel crucial. Atualmente, a epítome desta abordagem é considerar os **computadores como ferramentas mentais** (Jonassen, 2000). A sua relevância advém do facto que os computadores não conseguem ser muito bons a avaliar a aprendizagem das crianças, senão usando testes; e também porque as aplicações autónomas, não importa o quanto tenham êxito e sejam aprendíveis, podem não conseguir gerar empatia com uma criança ou situação específica. De facto, software adequado de uso autónomo para uma finalidade específica pode nem sequer existir. Já um contexto educativo baseado no professor pode ser adaptado às necessidades de uma criança específica.

Figura 57 – David Jonassen
De: https://courses.worldcampus.psu.edu/welcome/insys446/images/Jonassen.jpg

Utilizar computadores como ferramentas mentais significa utilizar ferramentas de software para "*funcionarem como parceiros intelectuais do aprendiz, para o envolver e promover o pensamento crítico e aprendizagem de nível elevado*" (Jonassen, 2000, p. 9). O que isto significa é que segundo esta perspetiva o software informático se divide em dois grupos distintos: ferramentas de produtividade e ferramentas mentais. Utilizar software como ferramenta de produtividade é usá-lo para simplificar a produção de algum artefacto, mas não mudra radicalmente a forma como é criado. Os processadores de texto, por exemplo, são um caso comum de uma ferramenta de produtividade: permite pegar num texto e apagá-lo, reformatá-lo, redimensioná-lo; avisam de potenciais erros ortográficos; apresentam uma visão da organização do texto numa página impressa; automatizam tarefas monótonas como criar índices ou atualizar referências internas. Mas para muitas – senão mesmo para a maior parte – das pessoas, não altera a forma fundamental de pôr as palavras por escrito.

[8] No seu livro de 2003, James Paul Gee parte deste conceito para deduzir, analisar e apresentar 36 "princípios de aprendizagem" que se podem extrair dos videojogos.
[9] Alguns professores empregam nas suas aulas videojogos que não foram criados como edutenimento. Por exemplo, pode-se obter um relato em primeira mão com alunos de faculdade no trabalho de Amory et. al. (1999).

> *«* (…) *os processadores de texto tornaram-nos a todos escritores mais eficientes, mais eficazes e mais produtivos. O que é discutível é se* (…) *os processadores de texto nos tornaram escritores* ***melhores****.* (…) *Certamente facilitam o processo* (…) *mas não é garantido que amplifiquem o processo. Não estou convencido que os romances de William Faulkner* (…) *teriam sido melhorados significativamente se Faulkner tivesse usado um processador de texto em vez da sua máquina escrever Royal manual. »*
>
> *(Jonassen, 2003, p. 16)*

Evidentemente, nem todas as tarefas de processamento de texto são de teor literário; e mesmo para essas esta prespetiva é talvez demasiado rígida. Para uma tarefa banal de escrita, feita por um escritor banal, é provável que uma característica simples como "procurar" possa melhorar imenso o produto da escrita. Considerando que qualquer linha de diálogo de um personagem pode ser antecedida por algo como "[Mordomo]", por exemplo, um escritor pode rapidamente verificar todas as frases do personagem e mais facilmente afinar o seu modo de falar. Ou encontrar lugares e circunstâncias onde um adjetivo específico foi usado – algo que exigiria uma memória prodigiosa a um escritor tradicional. Isto é apenas um exemplo de como uma ferramenta simples como um processador de texto pode afetar imensamente a ação do utilizador. Mas se estes impactes são ou não "radicais" é discutível. Além disso, esta observação não altera o argumento de que algumas ferramentas, como os processadores de texto, são fundamentalmente dedicadas à melhoria e facilitação da execuação de tarefas.

Por contraste, as ferramentas mentais proporcionam formas alternativas de pensar. Isto é um pouco uma característica das próprias ferramentas, um pouco uma questão da forma como são usadas. Por isso, um exemplo ajuda a clarificar o que é uma ferramenta mental: uma folha de cálculo, por exemplo, pode ser usada como ferramenta de produtividade e como ferramenta mental (as Folhas de cálculo já foram mencionadas na secção "2.3.2 – Programação para utilizadores finais").

Numa folha de cálculo, um utilizador pode introduzir dados numéricos em células específicas de uma grelha. Geralmente, as folhas de cálculo são utilizadas para contabilidade, automatizando uma multiplicidade de cálculos. O utilizador pode introduzir, por exemplo, os preços de todos os produtos vendidos numa certa situação e obter a soma resultante noutra célula; se houver um erro num dos números introduzidos, basta editar essa célula e a soma resultante é atualizada instantaneamente. Com bastante frequência, esta capacidade para produção de cálculos sem esforço é utilizada para apoiar a tomada de decisões: por exemplo, em vez de adicionar valores conhecidos, podem-se introduzir vários valores diferentes na mesma célula, para analisar os correspondents impactes no resultado.

Quando se usa para automatizar cálculos, uma folha de cálculo é uma ferramenta de produtividade: aquilo que faz também podia ser feito à mão (e assim foi feito durante séculos). Contudo, quando se usa para analisar o impacte de variados valores, isto implica uma mudança significativa: o utilizador-contabilista já não é apenas uma calculadora, mas também um "*testador de hipóteses (jogano jogos "e se…")*" (Jonassen, 2000, p. 87).

Mas as folhas de cálculo dão ainda mais um passo em frente, porque a maior parte dos utilizadores tem de construir as suas próprias fórmulas. Quando uma célula de folha de cálculo (por exemplo, a situada na coluna A da linha 10) apresenta a soma de todas as células acima dela (coluna A, linhas 1 a 9), isso é porque alguém introduziu uma formula de somar nessa célula A10 (provavelmente, "=SOMA(A1:A9)"). Quando se trata de operações comuns, costuma-se desenvolver e vender aplicações personalizadas de contabilidade ou apoio à decisão, para que os utilizadores de computadores possam beneficiar deste comportamento, sem terem de saber como construir tais fórmulas. Por isso, as pessoas que usam folhas de cálculo, em vez de aplicações específicas de contabilidade (ou outras aplicações de processamento numérico) costumam acabar por recorrer a elas para as tarefas que possuam alguma particulariedade, alguma característica específica da situação do utilizador, para a qual não há uma aplicação de software específica. Isto significa que os utilizadores de folhas ´de cálculo têm de introduzir as suas próprias fórmulas, para produzirem os resultados de

cálculo desejados. Para o fazerem, eles "*envolvem-se numa variedade de processos mentais* (…) *para utilizer as regras existentes, gerar novas regras para descrever relações e organizer informações* (…). *A ênfase na construção de uma folha de cálculo está na identificação e descrição dessas Relações em termos de regras de nível elevado*" (Jonassen, pp. 87-88).

Assim, em resumo, a utilização de um software como ferramenta mental envolve utilizar as capacidades dele para obter uma compreensão mais aprofundada das matérias que se estuda ou com que se trabalha. Isto requer que os professores preparem atividades que impulsionem os alunos, para que usem o software dessa forma, analisando domínios de contéudo, conduzindo o estudo e orientando quanto ao uso do software, ajudando-os a planear, a adaptar modelos existentes de uso da ferramenta, a criar e completar especificações, a extrapolar e a refletir (Jonassen, 2000).

> *« As ferramentas mentais são ferramentas de representação do conhecimento, que usam aplicações informáticas como bases de dados, redes semânticas (mapas conceptuais informáticos), folhas de cálculo, Sistemas periciais, ferramentas de modelação de sistemas, micromundos, motores especializados de pesquisa informativa, ferramentas de visualização, ferramentas de publicação multimédia, ambientes de conversação ao vivo e conferências informáticas para envolver os aprendizes em atividades de pensamento crítico.* (…) *Elas podem ser usadas em todos os currículos escolares para envolver os aprendizes, para que pensem com profundidade no conteúdo do seu estudo. »*
>
> *(Jonassen, 2000, p. 19)*

Como nota final, é interessante constatar que há várias ligações entre o que Jonassen chama de "uso como ferramenta mental" e os casos de programação de computadores que apresentei na secção "2.3.3 – Programação por utilizadores finais". No seu livro do ano 2000, ele desiste de encontrar uma utilização dos processadores de texto como ferramentas mentais, deixando em aberto apenas a possibilidade do seu conceito de ferramenta mental ser menos inclusive do que o dos leitores. Mas creio que as atividades de definição de estilos para o texto, como as descritas na secção 2.3.2, se enquadram no conceito de ferramenta mental de Jonassen. E de facto, várias partes do que ele chama "uso como ferramentas mentais" pode ser associado às competências de programação de computadores.

Contudo, Jonassen opõe-se a que as linguagens de programação de computadores possam ser ferramentas mentais (as suas observações referem-se ao Logo, mas os motivos que apresenta aplicam-se a outras linguagens), por falharem numa das suas regras para a definição de ferramentas mentais, especificamente que devem ser fáceis de aprender:

> *« Embora o Logo seja uma linguagem sintaticamente simples, ainda assim requer vários meses de treino para se desenvolver as competências suficientes para criar facilmente micromundos. »*
>
> *(Jonassen, 2000, p. 156)*

Sucede que a criação de micromundos não é o objetivo do uso de linguagens de programação na educação, embora seja uma das metodologias para tal. Há bem mais do que um punhado de usos da programação de computadores que podem ser aprendidos e aplicados com eficácia em menos de 1 ou 2 horas, por isso esta objeção é questionável face à própria definição de Jonassen da sua regra de ser "facilmente aprendível", que aqui transcrevo na íntegra, para a poder discutir:

> *« O esforço mental necessário para aprender a utilizer o software não deve exceeder os benefícios para o pensamento que advêm dele. Se forem necessárias semanas de esforço para aprender a usar um software, esse software torna-se o objeto da aprendizagem, em vez das ideias que estão a ser estudadas. A sintaxe e o método de uso do software não devem ser nem*

> *tão formais nem tão difíceis que obscureçam o objetivo mental do sistema. Os programas de software que são demasiado complicados de usar não são boas ferramentas mentais. A funcionalidade básica do software deve poder ser aprendida em 1 a 2 horas. Pode querer que os estudantes pensem casualmente acerca de informação num domínio do conhecimento, mas se o sistema exige semanas de esforço agonizante até ser aprendido, os benefícios de pensar dessa forma serão esmagados pelo esforço de aprendizagem do sistema. »*
>
> *(Jonassen, 2003, pp. 18-19)*

Proponho que se analise a leitura e a escrita como ferramentas mentais, embora não sejam baseadas em computadores. Jonassen, indiretamente, reconhece-as como tal, quando ele classifica a linguagem como uma "tecnologia cognitiva" (2003, p. 13), ao discutir os atributos das ferramentas mentais. Mas não se consegue dominar a leitura nem a escrita em 1 ou 2 meses (para algumas crianças, nem sequer em 1 ou 2 anos) muito menos 1 ou 2 horas. E para algumas crianças, as palvras "esforço agonizante" certamente seriam uma apta descrição dos seus sentimentos ao esforçarem-se na escola para aprender a ler e a escrever. Então a leitura e a escrita não são ferramentas mentais, ou se o são, são más ferramentas mentais? Estarão elas a ser aprendidas só por o serem? Pelo contrário, estão a ser aprendidas (do ponto de vista do indivíduo) para evitar que as crianças sejam limitadas, na sua vida adulta, pelo analfabetismo e todas as suas consequências associadas (pessoais, profissionais, sociais, etc.).

A classificação da leitura e da escrtia como ferramentas mentais, proporcionando formas alternativas de pensar, é suportada pela investigação neurológica recente. Estudos comparativos entre adultos alfabetizados e analfabetos revelaram diferenças significativas no "*desempenho de tarefas específicas e, nalguns casos, no padrão de ativação functional do cérebro e no volume de certas regiões anatómicas*" (Castro-Caldas, 2003). Recentemente, isto foi ainda mais realçado quando resultados de investigação revelaram que os adultos que aprenderam a ler e a escrever durante a infância eram mais rápidos e precisos, a efetuar tarefas que não estavam relacionadas com a leitura nem com a escrita, do que adultos que só aprenderam a ler e a escrever na idade adulta.

> *« Relatam-se resultados relativos ao processamento visual, operações intermodais (audiovisuais e visuotáteis) e cruzamento interhemisférico de informações. Os estudos com magnetoencefalografia, tomografia por emissão de positrões e ressonância magnética functional forneceram evidências de que a ausência de frequência escolar durante a idade habitual constitui uma limitação para o desenvolvimento de certos processos biológicos que proporcionam função comportamental.* (…) *o lobo occipital processou informações mais lentamente nos casos que aprenderam a ler em adultos, comparativamente aos que aprenderam a ler na idade habitual. »*
>
> *(Castro-Caldas, 2004, resumo)*

A meu ver, a definição da regra de "facilmente aprendível" foi escrita com o objetivo específico de tentar excluir a programação de computadores, que já era usada na educação há bem mais de uma década quando Jonassen escreveu o seu livro sobre ferramentas mentais. A frase inicial, sobre ponderar esforço e benefícios, bastaria como definição perfeitamente razoável e capaz da definição de "facilmente aprendível".

Acho que estas preocupações incluem um certo viés da parte de Jonassen contra a aquisição de conhecimentos inerentes ao domínio da informática, algo que se torna evidente quando ele afirma que "*muitos mais estudantes são* [hoje] *capazes de usar computadores sem formação*" (Jonassen, 2000, p. 8), para poder ignorar a formação acerca do funcionamento dos computadores e do seu software – algo de que discordo profundamente, devido à minha experiência no ensino de informática a alunos de 18, 19 e 20 anos do curso de educadores de infância, se por "utilizar computadores" ele pretende dizer "usar computadores com eficácia". Embora muitos aspetos dos computadores sejam

de facto desnecessários à vida moderna, assim como aprender a usar uma pena de escrita ou aprender a caçar, talvez seja um pouco demais exigir que os sistemas informáticos sejam tão transparentes como um televisor ou uma torradeirs, particularmente devido às suas características de metaferramenta, expostas na secção 2.1 – "Body & Mind: hardware and software". A meu ver, não há que temer que se possa aprender algo acerca dos computadores enquanto os usamos para aprender outra coisa. Além disso, embora aprender uma linguagem de programação possa de facto ser algo tornado problemático e incómodo, também pode ser uma experiência excitante e apaixonante, que envolve profundamente os aprendizes.

> *« Recordando os milhares de horas em que vimos miúdos a programar em Logo, uma coisa se destaca: a diversão que alguns têm só a brincarem, nem sempre a explorar ou a criar, mas a brincarem com coisas que já fizeram. Todos vivenciámos o constraste: o cuidado penoso, de criar um quadrado, que seja, a exploração para encontrar os ângulos certos, as falsas partidas. Mas então, quando se roda o quadrado e se faz uma estrela, a atmosfera invariavelmente muda, algo fica vivo e a brincadeira faz emergir o puro divertimento de descuidadamente, ou melhor, despreocupadamente brincar com aquilo que acabou de ser criado.»*
>
> *(Hoyles & Noss, 1999)*

Que esta última situação não suceda sempre na experiência pessoal de muitas pessoas, é talvez o motive para a falta de apoio de Jonassen. Dou algum enquadramento histórico para apoiar esta opinião na próxima secção (2.4.5).

A linha de raciocínio que usei acima relativamente à leitura e à escrita pode também ser aplicada à maior parte do conteúdo curricular presente nas escolas, pelo que também há algum conflito entre a ênfase de Jonassen nas ferramentas mentais para aprender e o conteúdo que está a ser aprendido, que se for para se ser coerente devia ser escolhido como ferramentas mentais para a vida.

A exclusão da programação de computadores da classificação de "ferramenta mental" de Jonassen é um dos motivos (mas não o único) para esta ser apresentada nesta tese como um modelo separado do uso educativo de computadores: o uso da programação de computadores, descrito na próxima secção, "Aprender o pensamento, com computadores", tem profundas implicações naquilo que as pessoas aprendem e como o aprendem, justificando-se que tenha uma secção própria. (Após a qual retornarei a esta secção para fechar o círculo.)

2.4.5 Aprender o pensamento, com computadores

Este quarto modelo do uso educativo da informática tem duas ligações principais ao anterior, "Aprender com computadores":

- o principal objetivo do uso de computadores não é facultar informações ou conhecimento aos estudantes, mas antes ser usado para melhorar o processo de aprendizagem, permitindo aos estudantes explorar ou melhorar a sua ligação à matéria de estudo[10];
- o processo de aprendizagem pode envolver um processo de conceção e desenvolvimento, ou seja, um processo de construção de um produto ou artefacto real (não necessariamente físico).

A principal diferença entre estes modelos é que no modelo "aprender o pensamento, com computadores" a exploração de conceitos se centra inteiramente no cultivo de oportunidades para que os alunos percebam como é que pensam sobre os conceitos que têm em mãos. Este modelo pega no

[10] Kahn (s.d.) inclui o aprender o pensamento, o título desta secção como senão uma de várias competências de pensamento crítico que podem ser aprendidas através da programação de computadores: decomposição de problemas, composição de componentes, representação explícita, abstração e pensar sobre o pensamento.

lema de que a melhor forma de aprender uma matéria é ensiná-la, propondo uma solução para um grande problema com a implementação desse lema: ensinar quem?

A resposta deste modelo é "ensinar um computador". Ou seja, ensinando um computador a fazer algo, o objetivo é que as crianças não só aprendem acerca desse algo, como também acerca de como o computador pensa; e nesse processo, acerca de como elas próprias pensam. Esta observação pode ser enigmática para alguns leito0res. Afinal de contas, o estilo informático de "pensamento" é bastante diferente do estilo humano de pensamento. Segundo este modelo educativo, este constraste é visto como extremamente valioso. Como explico na secção 4.1.2 (p. 259), as crianças pequenas (e mesmo alguns adultos) são frequentemente egocêntricas, no sentido em que frequentemente não percebem ou não dão conta da existência de diferentes perspetivas e maneiras de pensar diferentes das suas próprias. "*Proporcionando um modelo muito concreto e terra-a-terra de um estilo particular de pensar, trabalhar com o computador* [programando-o] *pode fazer com que seja mais fácil compreender que existe tal coisa como um 'estilo de pensamento'*" (Papert, 1980, p. 27).

Outra diferença é que neste modelo a construção de um produto ou artefacto real vai além do nível "pode acontecer" do modelo anterior: aqui, esta construção é algo que está quase sempre integrado no processo de aprendizagem.

Historicamente, o uso de programação de computadores na educação iniciou-se no final da década de 1950, na sua maioria com alunos de bacharelato/licenciatura em engenharia; as abordagens variavam entre o uso de código-máquina ou linguagens *assembly* (*vd.* secção 2.2.5) e o uso das primeiras linguagens próximas do ser humano, como o Fortran (*vd.* secção 2.2.6). Em 1960, isto era considerado uma questão de opção entre linguagens de programação "*orientadas para o computador ou orientadas para o problema*" (Katz, 1960, p. 527). Já então o principal objetivo de ensinar programação era "(…) *usar os computadores melhor e mais sabiamente, em virtude de os compreender primeiro como ferramentas gerais a usar para raciocinar sobre soluções para problemas do que como aparelhos para resolver problemas específicos – porque, neste ultimo caso, os estudantes* (...) *podem não ser capazes de generalizar adequadamente a sua experiência*" (Alan Perlis, acc. to Katz, 1960, p. 522). Mas a primeira linguagem de grande disseminação que foi criada com finalidades educativas (e não matemáticas) foi o BASIC, desenvolvido em 1963/1964 por John Kemeny e Thomas Kurtz, no Dartmouth College, em Hanover, Nova Hampshire, EUA (Kemeny & Kurtz, 1984; Hauben, 1995).

> *« Dartmouth adquiriu o seu primeiro computador em 1959, um muito pequeno chamado LGP-30. Kemeny facilitou o uso do LGP-30 pelos alunos de bacharelato. O engenho e a criatividade de alguns dos estudantes, a quem tinha sido dada experiência prática, espantou os docentes de Dartmouth. Kemeny e Thomas Kurtz, também do departamento de matemática de Dartmouth, foram assim encorajados a "pôr em movimento o conceito revolucionário de disponibilizar computadores aos estudantes da faculdade tão livremente quanto os livros da biblioteca." (Portraits in Silicon, Robert Slater, Cambridge, 1987, p.22.) O objetivo era tornar acessível a todos os estudantes o maravilhoso ambiente de investigação que os computadores podiam proporcionar. »*
>
> *(Hauben, 1995)*

Figura 58 – Thomas Kurtz & John Kemeny
De: http://www.atariarchives.org/deli/kurtz_and_kemeny.jpg

Este objetivo de Kemeny e Kurtz levou ao desenvolvimento de um sistema de partilha do tempo informático, para que muitos estudantes pudessem ter acesso a um computador. Mas tal não seria

suficiente: à altura, os métodos e linguagens de programação de computadores eram muito abstratos e dependiam do funcionamento do equipamento físico. O seu desejo de tornar a programação de computadores acessível a grandes números de pessoas significava que a simplificação e generalização para várias plataformas de equipamento informático estavam na ordem do dia: a programação precisava de utilizável por pessoas para quem os computadores não eram o seu foco principal de estudo ou de trabalho. Isto livou à criação da linguagem BASIC.

> *« Os objetivos de conceção do BASIC incluiam a facilidade de aprendizagem para o programador iniciante ou ocasional, independência do equipamento informático e do sistema operativo, a capacidade para acomodar programas grandes escritos por utilizadores especializados e mensagens de erro sensatas em inglês. »*
>
> *(Kemeny & Kurtz, 1984)*

Fundamentalmente, o BASIC tornou-se a primeira linguagem de escolha para uso da programação na educação, devido a ter sido a primeira linguagem orientada a programadores não profissionais. Tinha uma sintaxe que tentava assemelhar-se à língua inglesa e tentava tamém ser executada independentemente do equipamento informático.

Contudo, ter sido a primeira linguagem deste género também foi uma fonte de problemas para o desenvolvimento do BASIC. À época, Kemeny e Kurtz não produziram uma implementação da linguagem que tivesse qualidade comercializável. E embora o Dartmouth College tivesse os direitos sobre o BASIC, disponibilizou-a gratuitamente (Hauben, 1995). Inicialmente, a linguagem e o seu sistema fundamental foram usados em "*campi universitários, entidades governamentais e situações militares*" (Hauben, 1995). A disseminação da linguagem deu-se quando surgiram os primeiros microcomputadores, mas as capacidades limitadas dos seus componentes forçou a que a maior parte das versões de BASIC fossem simplificadas tecnicamente (o que geralmente consistia em perder algum tipo de usabilidade humana). Teve particular êxito um subconjunto de comandos de BASIC escrito em 1975 por Bill Gates e Paul Allen para o computador Altair. A expansão explosiva do uso de microcomputadores durante a década de 1980[11] significou que a maior parte das pessoas que usava um computador tinha acesso a algum tipo de linguagem BASIC. No final dessa década, 10 a 12 milhões de crianças em idade escolar tinham aprendido BASIC (Hauben, 1995).

Ao longo de todo este tempo, a linguagem BASIC desenvolvida por Kemeny e Kurtz tinha evoluído, desde a sua forma primitiva, incorporando temas avançados de ciências da computação, que estavam a aparecer nas linguagens. Mas como cada fabricante de computadores tinha a sua própria versão de BASIC, que como referi acima era geralmente limitada devido às capacidades do equipamento (ou mera despreocupação), a linguagem BASIC utilizada pela maior parte das pessoas era apenas uma pálida imagem disto.

> *« E assim, o BASIC na nossa família era uma linguagem sofisticada, com uma boa coleção de elementos construtivos estruturados, gráficos sofisticados, todas as boas ferramentas de modularização, etc. Que grande*

[11] De 1982 para 2000, a percentagem de lares dos EUA com computadores pessoais cresceu de cerca de 2% (U.S. Bureau of the Census, 1988) para aproximadamente 51%. Este crescimento foi mais rápido nos anos mais recentes do que na década de 1980, quando este número cresceu de 2% para 15%. Contudo, essa década representa o primeiro período de crescimento explosivo (General Motors, 2003; U.S. Bureau of the Census, 1993). Contudo, nas escolas este crescimento é de facto assinalável na década de 1980: em 1981 só 18,2% das escolas públicas tinham microcomputadores, mas este valor já tinha crescido para 97% em 1989 (U.S. Bureau of the Census, 1993).
Estatísticas para Portugal e para a União Europeia:

- Lares portugueses com computadores em 1995: 11%; em 2002: 28% (INE 2002).
- Lares da União Europeia com computadores no ano 2000: 35% (INRA 2000).

> *diferença para aquilo que o resto do mundo tinha de usar.* (…) *Para dar uma ideias de como outros tinham prostituído a nossa bela linguagem* (…) *»*
>
> *(Kemeny & Kurtz, 1984)*

De facto, o BASIC que era usado nas escolas ao longo desses anos era alguma das versões limitadas que Kemeny e Kurtz descartavam como "BASIC de rua". E essas versões da linguagem revelavam claramente as suas origens no mundo da matemática e do cálculo diferencial:

> *« O BASIC fora concebido para um público específico, para substituir o FORTRAN. O público seria alunos de cálculo diferencial, cientistas ou engenheiros que precisavam de efetuar series complexas de cálculos repetidamente.* (…) *E as pessoas que não compreendiam algoritmos matemáticos ou científicos? E as pessoas que queriam usar o computador para programação não numérica?»*
>
> *(Solomon, 1986, pp. 90-91)*

Nesta citação, Solomon falava das versões da linguagem habitualmente encontradas e usadas nos microcomputadores. Pouco importava que o BASIC estivesse a evoluir nos laboratórios de Kemeny e Kurtz, incorporando funcionalidades como modularidade e comandos gráficos: a linguagem disponível não correspondia às necessidades de milhões de potenciais utilizadores (crianças professores e pais) pelos quais estava a ser usada em casa e na escola.

> *« Os professores estão a descobrir que o BASIC é de difícil utilização pelos seus alunos.* (…) *Um dos problemas que os professores têm ao ensinar as crianças a programar é que dão ênfase à aprendizagem do vicabulário e da gramática da linguagem de programação em vez de destacarem o processo de exploração de ideias. »*
>
> *(Solomon, 1986, p. 92)*

Este relato tem um paralelismo notável com as preocupações de Jonassen relativamente ao tempo e esforço envolvidos na aprendizagem de uma linguagem de programação, apresentadas no final da secção 2.4.4. Mas Jonassen (2000, p. 156) referia-se à linguagem Logo, não ao BASIC. É irónico que o Logo, que apresentarei mais adianta nesta secção, seja uma linguagem que foi desenvolvida e evoluiu com o objetivo específico de ser usada por crianças e não por utilizadores adultos não profissionais… e no entanto possamos encontrar relatos relativos ao Logo que são semelhantes à observação anterior de Solomon, embora a um nível mais elevado de análise.

> *« As crianças pequenas e os seus professores* [que usam Logo] *cedo atingem o teto do que lhes é possível fazer dentro de restrições razoáveis de tempo e aptidão.* (…) *o conjunto de ferramentas e metáforas apropriadas para navegar ao nível da interface não são adequados abaixo desse nível – para se descer, tem de se entrar num novo mundo de dificuldade arcana (geralmente textual) que está desligado da maior parte das coisas que tornavam o trabalho introdutório tão envolvente. Uma forma de contornar estes problemas tem sido o desenvolvimento de micromundos. »*
>
> *(Hoyles & Noss, 1999)*

Mas esta situação irónica dá-nos que pensar: à medida que os computadores evoluíram, trazer a programação para os utilizadores finais tem sido um esforço recorrente de vários Investigadores, técnicos e até amadores. Tenho encontrado, frequentemente, pessoas que consideram a programação de computadores uma atividade mental incrivelmente agradável, a ponto de tentarem ensinar programação (ou "mostrar como se programa") a amigos, colegas e filhos, como um impulse tão natural como mostrar-lhes um dos seus amores: livro, série televisiva, paisagem ou o que seja. Este extrato de uma mensagem de correio eletrónico que recebi é um exemplo claro disto:

> *«* (…) *Gastei as melhores horas de cada fim de semana a brincar com N, a criança que estou a introduzir a isto. Costumo fazer intervalos para outras tarefas, porque sou estudante e tenho também de fazer os meus próprios projetos de programação. Frequentemente, quando estou a programar, ele senta-se ao meu colo e "ajuda-me". A minha intenção é introduzi-lo ao toontalk*[12] (sic) *a partir deste fim de semana, informando-o de que tenho um projecto de programação em que tenho de trabalhar. Então vou gastar mais ou menos 20 minutos a "trabalhar" e a apresentar-lhe cada um dos elementos principais do sistema. Por exemplo, no primeiro dia provavelmente farei o processo de decoração da minha casa,* (…) *»*
>
> *(Berthold, 2004)*

Tem de se ter presente que as linguagens de programação orientadas para utilizadores não profissionais surgiram com o BASIC em 1964. Isto representa apenas 40 anos de desenvolvimento desta tecnologia. No caso dos automóveis com motores de combustão interna, 40 anos de desenvolvimento significa 1903 (Barach, 1999); no caso dos aviões autopropulsionados, 1943 (USCFC, s.d.). Naturalmente, o ritmo moderno do progresso é mais rápido do que foi ao longo da história humana, mas mesmo que comparemos isto à evolução da sua tecnologia-irmã de rápido desenvolvimento, os computadores, 40 anos de desenvolvimento desde o Z1 de Zuse só significa 1981. E os serviços automáticos de telemóveis celebraram o seu 40.º aniversário em 1988 (Farley, 2005). É contudo justo dar conta de que outras tecnologias tiveram uma aceitação mais rápida: os televisores eletrónicos fizeram os seus 40 anos em 1967 (Bennett-Levy, 2001), altura em que já estavam muito popularizados[13]; e o predecessor da Internet, a ARPANET, ficou on-line em 1969 (Hauben, 1997) há apenas 36 anos.

Programar um computador continua a ser uma tarefa que requer esforço e concentração para se levar a cabo para mais do que simples atividades de iniciação. Considero que isto não é senão aquilo com se deve contar, dado tratar-se de uma tarefa que requer um raciocínio profundo, não uma simples reprodução ou adaptação de um modo de pensar empregue pela maioria das pessoas no dia-a-dia.

A programação de computadores é feita com vários tipos de linguagens de programação, cuja história se apresentou nas secções 2.2.5 e 2.2.6 (uma apresentação e enquadramento técnico foram feitos no capítulo 1). A sua história é paralela à história dos computadores, no sentido em que uma tecnologia que foi desenvolvida para facilitar cálculos numéricos foi posta ao serviço da facilitação de atividades não numéricas quotidianas. A meu ver, tal evolução é absolutamente fenomenal, mas não consegue facilmente fugir às suas raizes estritamente numéricas e lógicas.

> *« Quando comecei a aprender sobre programação* (…) *a minha ideias de como isso funcionaria era que devia ser como ensinar alguém a efetuar uma tarefa.* (…) *Imagine-se o meu choque quando descobri como é que a maior parte dos programadores de computadores realmente faziam o seu trabalho.* (...) *Havia estas coisas chamadas "linguagens de programação" que não tinham muito a ver com aquilo em que realmente se estava a trabalhar.* (...) *Esperem lá, pensei, não é possível que esta abordagem à programação funcione, de forma alguma! Ainda estoua tentar consertá-la. »*
>
> *(Lieberman, 2001, p. 2)*

Esta Correspondência imperfeita entre a ferramenta e a função[14] tem consequências importantes para o uso de linguagens de programação na educação: aquilo que as linguagens de programação são

---

[12] Uma linguagem de programação descrita na secção 3.3.5, que empreguei nas minhas atividades de campo.
[13] Em 1970, nos EUS, 95,3% dos lares tinha pelo menos um televisor (U.S. Bureau of the Census, 1988, p. 561).
[14] Relativamente a este problema da relação ferramenta↔função, um investigador referiu-se-lhe como uma fissura, dizendo que "*para os novatos, esta fissure é tão larga como o Grand Canyon*" (Norman, 1986, citado por Smith *et al.*, 2001).

hoje em dia é certamente apenas um lampejo daquilo que virão a ser, não apenas do ponto de vista técnico mas também conceptual, ou seja, quanto ao tipo de enquadramento mental que a programação irá requerer ou possibilitar. Thomas Dwyer, um dos primeiros investigadores de educação que atendeu ao uso de programação de computadores para fins educativos, disse uma vez:

> *« (…) acho que tentar inovar, com sistemas de apoio que nem sequer tentam corresponder à sofisticação do aprendiz humano, deve ser visto não como uma consequência mas uma traição à abordagem humanista da educação. »*
>
> *(Dwyer, 1971, p. 100)*

Assim, ainda hoje há necessidade de que o educador que emprega a programação de computadores se esforce por adaptar, alterar e de facto recriar as ferramentas de programação que são usadas, para manter o foco do raciocínio de cada aluno no próprio raciocínio[15].

O papel educativo do computador, a este nível, é interpreter o raciocínio do estudante, expresso através de um programa, expondo o estudante à relatividade das suas afirmações, mas também aos seus equívocos ou caminhos não tidos em conta, que são necessários para atingir o objetivo pretendido. Outra forma de o dizer é que o computador pode apresentar ao estudante um reflexo das suas ideias, afirmações e atividades. Este reflexo pode assim ser usado pelo estudante para obter uma perceção sobre a sua própria ideias, afirmação ou atividade. Uma característica particular deste processo de raciocínio baseado no computador é que um computador pode iterar as ideias de uma pessoa milhares ou milhões de vezes, ajudando o estudante a aperceber-se ou atingir uma compreensão dos impactes de longo prazo ou de grande disseminação de uma ideia.

No parágrafo anterior, apresentei apenas um papel educativo para os computadores, mas usei duas formulações completamente diferentes. E é possível que outras formulações possam ser feitas. Estas formulações também podem ser encaradas como perspetivas diferentes sobre o uso dos computadores. E as perspetivas costumam ditar o estilo de atuação de quem as vê. A consequência é que a "correspondência imperfeita entre ferramenta e função" que mencionei anteriormente é afetada pelos diferentes estilos humanos de abordar a ferramenta e de a suar para atingir um fim (sendo que aqui a ferramenta é o uso das linguagens de programação). Estas duas relações complexas, **ferramena↔função** e **ferramenta↔estilo**, conduzem o bom e mau uso da programação de computadores como modelo educativo.

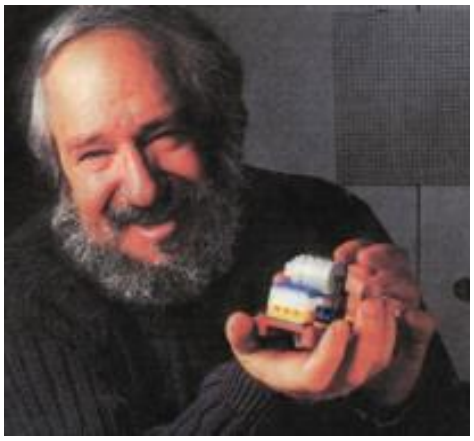

Figura 59 – Seymour Papert
De: http://www.papert.net/images/content/Papert-Leggos.JPG

Entre os pioneiros do uso educativo da programação de computadores, um dos primeiros a reconhecer esta relação complexa foi Seymour Papert. Antes de dedicar os seus interesses de investigação ao uso educativo dos computadores, ele trabalhou em Genebra com o epistemólogo suíço Jean Piaget[16] (Logo Foundation, 2000). Ele também levou a cabo investigação em inteligência

[15] Alan Kay afirmou, em 1984: "*O campo da informática ainda não teve o seu Galileu ou Newton, o seu Bach ou Beethoven, o seu Shakespeare ou Molière. Primeiro, precisa de um Guilherme de Occam*" (p. 43). Creio que isto também pode ser dito da informática educativa, mas que felizmente tanto os computadores como a informática educativa tiveram uma boa quota parte dos seus Pitágoras e Sócrates.

[16] It is therefore little wonder that he followed on Piaget's constructivist educational philosophy, presented in section **Erro! A origem da referência não foi encontrada.**. In his book *Mindstorms*, Papert states: "*I left Geneva enormously inspired by Piaget's image of the child, particularly by his idea that children learn so much without being taught*" (Papert, 1980, p. 215). But rather than being a mere follower, he provided new insights and contributions, under the urge noticeable from the continuation of the above quote; "*But I was also enormously frustrated by how little he could tell us about how to create conditions for more knowledge to be acquired by children through this marvelous process of 'Piagetian learning'*" (*id., ibid.*).

artificial, tendo fundado em 1964, juntamente com Marvin Minsky, o laboratório de inteligência artificial do MIT.

> *« Em 1964* (…) *o foco da minha atenção eram as crianças, a natureza do pensamento, descobrir como as crianças se tornavam pensadores. Mudei-me para o MIT* (…) *e agora as minhas preocupações imediatas eram com o problema da inteligência artificial: como fazer máquinas que pensam? »*
>
> *(Papert, 1980, p. 208)*

Enquanto trabalhava no MIT, Papert começou por se centrar no problema de adaptar a ferramenta à função. A função não era apenas instruir ou educar crianças, mas antes um corolário do seu percurso de investigação sobre o pensamento: "*ver ideias das ciências da computação não apenas como **instrumentos de explicação** de como a aprendizagem e o pensamento de facto funcionam, mas também como **instrumentos de mudança** que poderiam alterar e possivelmente melhorar a forma pela qual as pessoas aprendem e pensam*" (Papert, 1980, pp. 208-209).

Em vez de tentar usar as linguagens de programação então disponíveis, em 1966 e 1967 Papert concebeu uma nova linguagem de programação em colaboração com Wallace Feurzeig e Daniel Bobrow, da empresa Bolt Beranek and Newman, Inc., a que chamou Logo[17] (Feurzeig, 1984; Papert, 1980, p. 210; Logo Foundation, 2000). O Logo foi depois implementado por uma equipa liderada por Feurzeig e Bobrow, nessa mesma empresa[18] (Feurzeig, 1984; Papert, 1980, p. 210).

> *«* (…) *conceber uma linguagem de computador que fosse adequada para crianças. Isto não significava que devesse ser uma linguagem "de brincar". Pelo contrário, eu queria que tivesse o poder das linguagens de programação profissionais, mas também queria que tivesse caminhos de entrada fácil para iniciantes não matemáticos. »*
>
> *(Papert, 1980, p. 210)*

Uma forma diferentes de resumir estas ideias é que ele lançou-se a pegar nas melhores indeias das ciências da computação e fazer com elas engenharia de infância (Papert, 1977, in Kahn, 2001a).

> *« O nome LOGO foi escolhido*[19] *para a nova linguagem para sugerir o facto de que é em primeiro lugar simbólica e só em segundo lugar quantitativa. »*
>
> *(Papert, 1980, p. 210)*

O Logo é contemporâneo do uso dos computadores para CAI (secção 2.4.3) e do BASIC. Dadas as capacidades gráficas limitadas dos computadores na década de 1960, era uma linguagem baseada em comandos escritos, como toda as outras linguagens da época. Mas a sua finalidade específica, desde a sua fase de conceção inicial e daí para a frente, era ser usada por crianças programadoras. Isto fez com que várias das suas funcionalidades principais fossem planeadas, em vez de serem acrescentadas como remendos, para permitir ao programador centrar-se mais na ideia do programa e menos na sua implementação escrita (em termos de ciências da computação, gastar mais tempo a programar e menos a codificar).

> *« Hoje em dia, em muitas escolas, a frase "instrução apoiada por computador" significa fazer com que o computador ensine a criança. Poder-se-ia dizer que o computador está a ser usado para programar a criança. Na*

---

[17] Do grego antigo λόγος, que significa "pensamento de palavra" (Kypros-Net, n.d.).

[18] "*A conceção inicial do Logo surgiu através de longas discussões em 1966 entre Seymour Papert, Dan Bobrow eu próprio. Papert desenvolveu as especificações funcionais gerais da nova linguagem e Bobrow deu vastos contributos à conceção, além da primeira implementação. Subsequentemente, Richard Grant fez muitos acrescentos e modificaçoes à conceção e à implementação, apoiado por Cynthia Solomon, Frank Frazier e Paul Wexelblat. Eu dei ao Logo o seu nome*" (Feurzeig, 1984).

[19] Por Wallace Feurzeig (*vd.* nota de rodapé 55, na página 15).

> *minha visão, a criança programa o computador e, ao faze-lo, adquire um sentido de mestria sobre uma peça da mais moderna e poderosa tecnologia e estabelece um contacto íntimo com algumas das ideias mais profundas da ciência, da matemática e da arte da construção de modelos intelectuais. »*
>
> *(Papert, 1980, p. 5)*

Como exemplos de tais funcionalidades apontem-se a simplificação da sintaxe e da semântica, mensagens de erro bem concebidas, uma definição transparente de procedimentos e a interpretação direta do código.

Desde 1967 e ao longo da década de 1970, Papert esteve envolvido na programação de Logo por crianças, em diversos ambientes educativos associados à investigação[20], dentro e for a da escola, com algumas crianças pequenas, crianças mais velhas e adultos. Graças aos seus interesses anteriores na educação e no pensamento, este envolvimento de Papert consolidou as suas ideias acerca do significado educativo da programação de computadores, que obtiveram pela primeira vez disseminação geral graças ao seu livro *Mindstorms* (1980) – em português "Logo: Computadores e Educação".

Nesse livro, Papert centrava-se nas suas ideias educativas, a filosofia por trás do uso de uma linguagem de programação como o Logo. Apresento as suas ideias educativas na secção 4.2.2 – "Seymour Papert and constructionism", mas a bem da clareza mencionarei aqui algumas. Entre elas: **aprendizagem sintónica** (Papert, 1980, p. 63) ou aprender em sintonia com o aprendiz, isto é, quando a aprendizagem se relaciona com o sentido e conhecimento da criança acerca do seu próprio corpo (corpo-sintónica) ou é coerente com o sentiment da criança acerca de si, dos seus gostos, antipatias, objetivos e intenções (ego-sintónica); **conhecimento procedimental** ou o uso da programação como material para pensar e falar acerca de procedimentos – todo o tipo de procedimentos, não apenas procedimentos informáticos (Papert, 1980, p. 223); **micromundos** como "incubadoras de conhecimento", no sentido de ambientes com conteúdo limitado, para que os seus princípios possam ser simulados, vivenciados e analisados (Papert, 1980, cap. 5, pp. 120-134); e a **depuração** (*debugging*) como forma de instigar a noção de que algo que não corre bem é uma oportunidade "*para estudar o que aconteceu, para compreender o que correu mal, e através dessa compreensão resolve-lo*" (Papert, 1980, p. 114).

No final da década de 1970, o público do Logo começou a crescer, com o surgimento dos microcomputadores: até então, tinha estado confinado aos locais de investigação e a um punhado de escolas. Os projetos-piloto lançados em 1980 envolveram centenas de crianças e extensa formação de professores. Ao longo da década de 1980, o uso de Logo aumentou e foram introduzidas novas versões, de variadas editoras (Logo Foundation, 2000).

Durante este período, a filosofia Logo e a linguagem de programação beneficiaram da exposição a um grande número de crianças e de utilizadores adultos, tendo Papert centrado-se mais no problema dos diferentes estilos de utilização da ferramenta: o problema ferramenta↔estilo.

Já no seu livro de 1980 ele tinha exposto várias ideias acerca dos padrões individuais e dos estilos de raciocínio (*e.g.*, Papert, 1980: cap. 1, sobre computadores e culturas informáticas, pp. 19-37; estilos para aprender competências "físicas" e "intelectuais", pp. 104-105; programação com e sem estrutura, pp. 103-104). Mas por essa altura, ele ainda não tinha desenvolvido completamente estas ideias[21].

Em 1990, no entanto, ele e Sherry Turkle publicaram um artigo sobre o confronto de estilos de programação preferidos por pessoas diferentes (Turkle & Papert, 1990). Nele, contrastavam a

[20] Os locais de investigação situavam-se no MIT (EUA), Edimburgo (Escócia) e Tasmânia (Austrália) (Logo Foundation, 2000).

[21] Cf. comentários de Papert em 1980 (p. 104) relativos às abordagens à programação estruturada e a sua perspetiva sobre o assunto em 1990 (Turkle & Papert, 1990, pp. 138-140).

abordagem formal à programação e à resolução de problemas, de cima para baixo, com a abordagem informal, de engenhocas, de baixo para cima.

O que Turkle e Papert perceberam foi que a cultura do informáticos, como a cultura da Ciência, da matemática e da educação em geral, favorecia a abordagem formal, de cima para baixo, à resolução de problemas: um problema seria dividido em várias partes mais pequenas, sendo depois cada uma destas atacada de forma independente, criando um "procedimento" que daí para a frente seria uma caixa preta, ou seja, uma unidade atómica, indivisível, a ser usada e cujo funcionamento interno não é nunca tido em conta.

> *« Os programadores estruturados geralmente não se sentem confortáveis com uma construção até estár completamente reduzida a caixas pretas, com todo o seu funcionamento interno e todos os vestígios do processo – talvez confuse – da sua construção ocultos da vista. »*
>
> *(Turkle & Papert, 1990, p. 140)*

Embora esta seja uma abordagem boa e eficiente de resolver problemas grandes, eles contrapõem que não é um método universal com o qual toda a gente se possa sentir confortável. Dos trabalhos do antropólogo francês Claude Lévi-Strauss, apropriaram-se do termo "*bricolage*" para descrever a abordagem que constrastava com a formal (*id.*, pp. 129, 135-143).

> *« Lévi-Strauss usou o termo "bricolage" ao contraste à metodologia analítica da ciência ocidental, a que ele chamou de "ciência do concreto" das sociedades primitivas. Os bricoleurs que ele descreve não se deslocam de forma abstrata e hierárquica de um axioma para um teorema até ao corolário. Os bricoleurs constroem teorias arrumando e rearrumando, negociando e renegociando um conjunto de materiais bem conhecidos. »*
>
> *(id., pp. 135-136)*

Ao contrastar estes estilos de pensamento aplicados à programação, eles fazem notar que os reais praticantes da ciência geralmente empregam aspetos do bricolage, o que pode pôr em causa se deve ser catalogado como inferior ao pensamento formal; e fazem analogias entre os programadores "bricolage" e pintores, cozinheiros, escultores e escritores que começam a escrever sem uma estrutura geral. Eles defendem a aceitação deste estilo de pensamento, em vez de o ostracizar como inferior: "O b*ricolage é uma forma de organizar o trabalho. Não é um estágio de progressão para uma forma superior*" (*id.*, p. 141).

Estas ideias integraram a filosofia educativa de Papert através dos conceitos de **proximidade** ao objeto de estudo, ou seja, quando o estudante se coloca a si mesmo no lugar desse objeto, e de **permissividade exigente**, ou seja, fazer com que seja claro que as crianças têm que se esforçar, mas com ampla flexibilidade na escolha de projeto (Papert, 1993, p. 124). Mas a sua principal consequência foi contribuir para a visão global de Papert, pela qual o atual sistema educativo escolar não pode ser reformado. Em vez disso, ele defende a necessidade de uma reconceção completa da forma pela qual a humanidade ensina as suas crianças.

> *«* (…) *imaginamos a emergência de uma ciência do pensamento que reconheça o concreto* [e não o formal] *como o seu objeto central. »*
>
> *(id., p. 155)*

Esta ideias já estavam de certa forma a ser expressas por Papert em 1980, sob a forma de um desconforto geral com a escola tradicional. Por ex.:

> *« Estas experiências* [de programação Logo] *exprimem uma crítica da matemática escolar tradicional (que se aplica tanto à assim chamada nova matemática como à antiga). Uma descrição da matemática escolar tradicional em termos dos conceitos que desenvolvemos neste ensaio revelá-*

> *la-ia como uma caricatura da matemática, na sua encarnação impessoal, puramente lógica, "formal". Embora possamos documentar a evolução na retórica dos professores de matemática (os professores da nova matemática são ensinados a falar em termos de "compreensão" e "descoberta"), o problema mantém-se por causa do que estão a ensinar. »*
>
> *(Papert, 1980, p. 205)*

As ideias educativas de Papert, que aqui se discutem, ganharam forma global através de uma nova filosofia educativa chamada **construcionismo**; foi apresentada pela primeira vez em 1991 (Papert & Harel, 1991). Uma tentative de a sintetizar (embora sem as ideias importantes acima referidas) é dizer que a construção do conhecimento pessoal é particularmente apoiada quando o aprendiz está envolvido na construção de artefactos com significado pessoal (Papert, 1999). O construcionismo é apreesntado na secção 4.2.2, "Seymour Papert and constructionism".

Para Papert, a consequência desta linha de raciocínio é que o sistema escolar atual é desadequado e não pode ser reformado, mas antes radicalmente alterado:

> *« Não se pode passar de uma carruagem para um avião a jato através de uma série de pequenas melhorias. Será possível ir da escola de ontem para a escola do amanhã através de uma série de melhorias incrementais?* (...) *enquanto as escolas confinarem a tecnologia à simples melhoria do que estão a fazer, em vez de realmente mudra o sistema, nada de muito significativo acontecerá. »*
>
> *(Papert, 1998)*

O construcionismo é também por vezes chamado, indiretamente de "filosofia Logo", mas parte desta lógica que tenho vindo a apresentar é que a linguagem de programação Logo, em si, é apenas uma tecnologia: um passo na direção certa, mas não um resultado definitivo. O nome Logo pode assim ser frequentemente encontrado para se referir às ideias educativas de Papert, representando uma filosofia educativa, não apenas uma linguagem de programação. As citações que se seguem mostram precisamente isto, visto que a primeira é de 1982, nove anos antes da publicação do livro "Constructionism" (Papert & Harel, 1991) e a segunda, além de ser do próprio Papert é de 1999, oito anos depois da publicação de "Constructionism".

> *« O Logo é o nome de uma filosofia educativa e de uma família de linguagens de programação em continua evolução, que ajudam à sua concretização. »*
>
> *(Abelson, 1982 apud Logo Foundation, 2000)*

> *« A linguagem de programação Logo está longe de ser tudo o que interessa e em princípio podemos imaginar usar uma linguagem diferente, só que a própria programação é um elemento-chave desta cultura. »*
>
> *(Papert, 1999, p. xv)*

Foram desenvolvidas várias alternativas modernas ao Logo. Podem ser usadas (e foram-no) no âmbito da filosofia Logo, sendo apresentadas nas secções 3.3.4 e 3.3.5. Entre elas:

- Stagecast Creator (anteriormente Cocoa, anteriormente KidSim), orientada para o desenvolvimento de simulações animadas e jogos com base em regras (uma descrição das atividades e uma discussão de teor educativo constam de Lewis *et al.*, 1997);
- Squeak, uma implementação de uma linguagem de programação orientada a objetos, o Smalltalk-80, que se pretende muito portátil, incluindo suporte independente da plataforma para cores, sons e processamento de imagem; a sua comunidade gerou os Squeak Etoys, "*ambientes computacionais que ajudam a pessoas a aprender ideias, construindo e brincando com eles*" (Kay, n.d.);

- ToonTalk, com o objetivo de proporcionar uma nova metáfora para descrever programas, utilizando personagens animadas: não há código "estático", todo o programa é uma animação (a sua relação com a filosofia e cultura Logo é discutida por Kahn, 2001a).

Uma perceção final é que visto que um aspeto crucial é a construção de artefactos, isto é algo que se pode atingir não apenas na programação, mas também na **conceção de artefactos**: "*A teoria do construcionismo sugere uma forte ligação entre a conceção e a aprendizagem: afirma que atividades que envolve o fazer, o construir ou o programar – em suma, o conceber – proporcionam um contexto rico para a aprendizagem*" (Kafai & Resnick, 1996, "Introduction"). Cada artefacto concebido existe for a da mente do aprendiz e tem de **passar pelo escrutínio do mundo físico, cultural e demais restrições da realidade**: torna-se algo que "*se pode mostrar, discutir, examiner, sondar e admirar. Fica exposto*" (Papert, 1993, p. 142). Estas perceções fornecem a ligação entre esta secção, "Aprender o pensamento, com computadores" e a anterior "Aprender com computadores". E desta forma, como então se prometeu, se fechou um círculo entre elas.

> *« O argumento* (…) *não é que* (…) *a programação de computadores, em geral, seja única enquanto propiciadora de um ambiente para aprender estas competências de pensamento.* (…) *Mas é um ambiente muito rico onde estes tipos de competências de pensamento são "exercitadas" com frequência num contexto natural. »*
>
> *(Kahn, s.d.)*

Referências

Abelson, Harold (1982). *Apple Logo*, ISBN 0070004250, Byte Books, Peterborough, NH, USA.

Amory, Alan; Naicker, Kevin; Vincent, Jacky; Adams, Claudia (1999). *The use of Computer Games as an Educational Tool: 1. Identification of Appropriate Game Types and Game Elements*, in "British Journal of Educational Technology", ISSN 0007-1013, vol. 30, no. 4, pp. 311-322, British Educational Communications and Technology Agency, Coventry, UK. Referenced from the on-line version, retrieved on January 27th, 2004, from http://www.nu.ac.za/biology/staff/amory/bjet 30.rtf.

Barach, John (1999). *The Origin of the Automobile*, in "Automobile History", Web site at http://www.motorera.com/history/history.htm, last accessed on May 18th, 2005.

Bennett-Levy, Michael (2001). *1935-1941 Timeline*, Web page found at the address http://www.tvhistory.tv/timeline1.htm, part of the Web site "Television History – The First 75 Years", located at http://www.tvhistory.tv/. Last accessed on May 18th, 2005.

Berg, Gary A. (2003). *The Knowledge Medium: Designing Effective Computer-Based Learning Environments*, ISBN 1-59140-103-8, Information Science Publishing, Idea Group Inc., Hershey, PA, USA.

Berthold, Frank (2004). *Re: Young Toon-Talkers*, e-mail message sent to Leonel Morgado on July 21st, 2004, at 01:23:17 -0400.

Bush, Vannevar (1945). *As We May Think*, in The Atlantic Monthly, July 1945. Referenced from the electronic version at http://arti.vub.ac.be/cursus/2001-2002/ai2/material/bush.pdf, retrieved on March 24th, 2003.

Castro-Caldas, Alexandre (2003). *Como encontrar áreas de interesse para estudar o cérebro analfabeto*, in Revista Psychologica 34, 2003, Faculdade de Psicologia e Ciências da Educação da Universidade de Coimbra, Coimbra, Portugal. Also referenced is version in English (Castro-Caldas, 2004), which includes updated research information.

Castro-Caldas, Alexandre (2004). *Targeting regions of interest for the study of the illiterate brain*, in International Journal of Psychology, February 2004, vol. 39, no. 1, pp. 5-17, ISSN 0020-7594, Psychology Press, Taylor & Francis Group, London, UK.

Dear, Brian L. (2004). *What's New at PLATOPEOPLE.COM*, Web page retrieved from http://www.platopeople.com/whatsnew.html on July 30th, 2004.

diSessa, Andrea A. (2000). *Changing Minds: computers, learning and literacy*, ISBN 0-262-04180-4, MIT Press, Cambridge, Massachusetts, USA.

Dozen, Patty (1998). *PLATO/NOVANET History*, e-mail message sent from the e-mail address pdozen@sunny.vcccd.cc.ca.us, on March 12th, 1998, at 12:01:15 -0800 to the mailing list "Open Forum for Learning Assistance Professionals", at e-mail address <lrnasst@listserv.arizona.edu>. Referenced from the on-line archived version, retrieved from http://www.lists.ufl.edu/cgi-bin/wa?A2=ind9803&L=lrnasst-l&F=&S=&P=16965 on July 30th, 2004.

Druin, Allison (1996). *CD-ROM Edutainment*, in "Designing Multimedia Environments for Children", ch. 2, pp. 62-93 Allison Druin, Cynthia Solomon, eds., ISBN 0-471-11688-2, John Wiley & Sons, Inc., New York, NY, USA.

Dwyer, Thomas A. (1971), *On the importance of complexity in supportive systems for educational computing*, Interface v. 5, no. 3: 99-105. Referenced from the on-line version of ACM SIGCUE Outlook, ISSN 0163-5735, ACM Press, New York, NY, USA, retrieved from http://doi.acm.org/10.1145/965836.965838 on August 4th, 2004.

Farley, T. (2005). *The first automatic radiotelephone service*, in "Mobiles Phones – A History", Web page at http://www.galaxyphones.co.uk/mobile_phones_history06.asp, Galaxy Phones, Wigan, Lancashire, UK. Last accessed on May 18th, 2005.

Feurzeig, Wallace (1984). *The Logo Lineage*, in "Digital Deli – The Comprehensive, User-Lovable Menu of Computer Lore, Culture, Lifestyles and Fancy by The Lunch Group & Guests", Steve Ditlea, ed., ISBN 0894805916, Workman Publishing Company, Inc., New York, NY, USA. Referenced from the on-line version, retrieved from http://www.atariarchives.org/deli/logo.php on August 31st, 2004.

Firaxis Games (sem data). *The Civilization III Series*, Web page, Firaxis Games, Hunt Valley, MD, USA. Retrieved on July 23rd, 2004 from http://www.firaxis.com/games_civ3.cfm.

Gee, James Paul (2003). *What Video Games Have To Teach Us about Learning and Literacy*, ISBN 1-4039-6538-2, Palgrave MacMillan, New York, NY, USA. Referenced from the paperback edition, edited in May 2004.

General Motors (2003). *Sales & Market Analysis Media Briefing*, presentation given by Paul Ballew, Executive Director of Market & Industry Analysis, in December 15th, 2003. Referenced from http://media.corporate-ir.net/media_files/IROL/84/84530/presentations/GM_121503_PB.pdf, on-line document retrieved on August 4th, 2004.

Hauben, Jay Robert (1995). *John G. Kemeny: BASIC and DTSS: Everyone a Programmer*, in "Computer Pioneers", J. A. N. Lee, ed., IEEE Computer Society Press, Los Alamitos, California, USA. Referenced from the on-line version, retrieved on August 3rd, 2004, from http://dtss.dartmouth.edu/people2.php.

Hauben, Michael (1997). *Behind the Net: The Untold Story of the ARPANET and Computer Science*, in "Netizens – On the History and Impact of the Net", Michael Hauben & Ronda Hauben (eds.), ISBN 0-8186-7706-6, ch. 7, pp. 96-114, IEEE Computer Society Press, Los Alamitos, CA, USA. Referenced from the online version at http://www.columbia.edu/~rh120/ch106.x07, retrieved on May 18th, 2005.

Hoić-Božić, Nataša (1997). *Hypermedia Supported Education*, M.Sc. Thesis, Fakulteta za računalništvo in informatiko (Faculty of Computer and Information Science), Univerza v Ljubljani

(University of Ljubljana), Ljubljana, Slovenia. Referenced from the electronic version, retrieved from http://top.pefri.hr/mr/, on June 28th, 2004.

Hoyles, Celia and Noss, Richard (1999). *Playing with (and without) words*, "Proceedings of the Seventh European Logo Conference Eurologo '99, Sofia, Bulgaria, 22-25 August 1999", edited by Nikolov, Roumen; Sendova, Evgenia; Nikolova, Iliana and Derzhanski, Ivan. ISBN 954-9582-03-5, Virtech Ltd., Sofia, Bulgaria. An on-line version is available (last checked on August 9th, 2004) at http://www.ioe.ac.uk/playground/RESEARCH/papers/playword.htm.

Hutchinson, Jamie (2003). *Don Bitzer and Gene Slottow: A PLATO-nic relationship thrived in ECE*, in "Ingenuity", Vol. 8, Number 1, May 2003, a newsletter for Electrical and Computer Engineering's student, staff, and faculty at the University of Illinois at Urbana-Champaign. Referenced from the on-line version at http://www.ece.uiuc.edu/ingenuity/503/PLATO.html, retrieved on July 30th, 2004.

INE (2002). Inquérito à Utilização de Tecnologias da Informação e da Comunicação pelas Famílias (2001 e 2002), Instituto Nacional de Estatística, Lisboa, Portugal. Referenced from the on-line version, retrieved from http://www.ine.pt/prodserv/quadros/163/179/005/xls/01000000.xls on August 4th, 2004.

INRA (2000). *Measuring Information Society 2000*, European Commission, Directorate General "Information Society", Brussels, Belgium. Referenced from the on-line version, retrieved from http://europa.eu.int/ISPO/basics/measuring/eurobaro/eurobaro53/docs/mis2000_report.doc on August 4th, 2004.

Jonassen, David H. (2000). *Computers as Mindtools for Schools*, 2nd ed., Prentice-Hall, Inc., Pearson Education, New Jersey, NJ, USA.

Kafai, Yasmin B. and Resnick, Mitchel (1996). *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*, ISBN 0-8058-1985-1, Lawrence Erlbaum Associates Inc., Mahwah, NJ, USA. Referenced from the on-line version, retrieved from http://www.gseis.ucla.edu/faculty/kafai/faculty/Book_CIP_Intro.html on August 9th, 2004.

Kahn, Ken (2001a). *ToonTalk and Logo – Is ToonTalk a colleague, competitor, successor, sibling, or child of Logo?*, in "Proceedings of EuroLogo 2001", Paedagogische Akademie der Dioezese Linz, Linz, Austria. Referenced from the on-line version, retrieved on August 6th, 2004, from http://www.ocg.at/activities/books/volumes/band%20156/K11Kahn.doc.

Kahn, Ken (sem data). *Thinking Skills*, research paper from the Playground project, on-line at http://www.ioe.ac.uk/playground/research/papers/thinking.htm, retrieved on November 23rd, 1999. It is also on-line at http://www.toontalk.com/English/skills.htm (last checked on August 9th, 2004).

Katz, Donald L. (1960, ed.). *Conference report on the use of computers in engineering classroom instruction*, in "Communications of the ACM", ISSN 0001-0782, vol. 3, no. 10, pp. 522-527, ACM Press, New York, NY, USA. Referenced from the on-line version, retrieved on August 31st, 2004, from http://doi.acm.org/10.1145/367415.993453.

Kay, Alan (1984). *Computer Software*, in "Scientific American", ISSN 0036-8733, vol. 251, no. 3, pp. 41-47, Scientific American, Inc., New York, NY, USA.

Kay, Alan (sem data). "*Etoys and SimStories in Squeak*", on-line article in applet, from http://www.squeakland.org/project.jsp?/projects/etoys/EtoysSimstories.004.pr, retrieved on August 6th, 2004.

Kemeny, John G.; Kurtz, Thomas E. (1984). *Bringing Up BASIC*, in "Digital Deli – The Comprehensive, User-Lovable Menu of Computer Lore, Culture, Lifestyles and Fancy by The Lunch Group & Guests", Steve Ditlea, ed., ISBN 0894805916, Workman Publishing Company, Inc., New York, NY, USA. Referenced from the on-line version, retrieved on August 3rd, 2004 from http://www.atariarchives.org/deli/basic.php.

Kypros-Net (sem data). *ΛΕΞΙΚΟ - LEXICON: Greek-English-Greek dictionary*, on-line resource at http://www.kypros.org/cgi-bin/lexicon, Kypros-Net Inc., Cambridge, Massachusetts, USA. Accessed on August 5th, 2004.

Lewis, Clayton; Rader, Cyndi; Brand, Cathy; Carlone, Heidi (1997). *Models Children Build: Content, Logic and Educational Impact*, paper presented at the Annual Meeting of the National Association for Research in Science Teaching (NARST) in April 1997, in Chicago, IL, USA. Referenced from the on-line version, at http://www.cs.colorado.edu/~crader/NARST97/ModAnl-Narst97.html, retrieved on July 10th, 2001.

Lieberman, Henry (2001). *Introduction*, in "Your Wish Is My Command: Programming By Example", Henry Lieberman, ed., ISBN 1-55860-688-2, Morgan Kaufmann Publishers, San Francisco, California, USA.

Logo Foundation (2000). *What is Logo?*, Web page at http://el.media.mit.edu/logo-foundation/logo/index.html, Logo Foundation, Epistemology and Learning group, The Media Laboratory, Cambridge, MA, USA. Retrieved on August 5th, 2004.

McDonald, Jason K.; Yanchar, Stephen C.; Osguthorpe, Russel T. (2005). *Learning from Programmed Instruction: Examining Implications for Modern Instructional Technology*, Educational Technology Research and Development, ISSN 1042-1629, vol. 53, issue 2, pp. 94-89, Association for Educational Communications and Technology, Bloomington, IN, USA. Referenced from the electronic version, retrieved from http://cid.byu.edu/mcdonald/pi.pdf on July 2nd, 2004.

McNeil, Sara (sem data). *A hypertext history of instructional design*, Web site hosted at the College of Education of the University of Houston, retrieved on July 1st, 2004, from http://www.coe.uh.edu/courses/cuin6373/idhistory/index.html.

Minsky, Marvin (1988). *The Society of Mind*, ISBN 0671657135, Simon & Schuster, New York, NY, USA.

NMAH-BC (2002). *Slates, Slide Rules, and Software – Teaching Math in America*, an exhibit at the National Museum of American History, Behring Center, in Washington, USA. Cited from the exhibition's Web site at http://www.americanhistory.si.edu/teachingmath/index.htm, retrieved on July 2nd 2004.

Ong, James; Ramachandran, Sowmya (2003). *Intelligent Tutoring Systems: Using AI to Improve Training Performance and ROI*, Technical Document, Stottler Henke Associates, San Mateo, CA, USA. Referenced from the on-line version, retrieved on August 18th, 2005, from http:// www.stottlerhenke.com/papers/ITS_using_AI_to_improve_training_performance_and_ROI.pdf.

Oppenheimer, Todd (2003). *Meet The Characters*, Web page of the companion site to the book "The Flickering Mind: The False Promise of Technology in the Classroom and How Learning Can Be Saved", retrieved from http://www.booknoise.net/flickeringmind/characters/ on July 27th, 2004.

Ozzie, Ray (2002), *Speaking Mind to Mind*, article in "The New York Times", Business section, December 1st, 2002, written with Glenn Rifkin. Referenced from the on-line version at http://www.nytimes.com/2002/12/01/business/yourmoney/01BOSS.html?ex=1091332800&en=155bc2ae7014f7c7&ei=5070, retrieved on July 30th, 2004.

Papert, Seymour (1977). MIT Logo Project meeting notes, referenced by Ken Kahn (2001).

Papert, Seymour (1980). *Mindstorms: Children, Computers, and Powerful Ideas*, ISBN 0-465-04629-0, Basic Books, New York, USA. Referenced from the second edition, ISBN 0-465-04674-6, Basic Books, New York, USA, 1993.

Papert, Seymour (1993). *The Children's Machine: rethinking school in the age of the computer*, ISBN 0-465-01063-6, Basic Books, New York, USA, 1993.

Papert, Seymour (1998). *Technology in Schools: To Support the System or Render it Obsolete*, on-line article published by the Milken Exchange on Education Technology, Milken Family Foundation, Santa Monica, California, USA. Retrieved on March 19th, 2004, from http://www.mff.org/edtech/article.taf?_function=detail&Content_uid1=106.

Papert, Seymour (1999). *Introduction: What is Logo? And Who Needs It?*, in "Logo Philosophy and Implementation", ISBN 2-89371-494-3, LCSI, Highgate Springs, Vermont, USA. Electronic book retrieved from http://www.microworlds.com/company/philosophy.pdf on March 30th, 2004.

Papert, Seymour; Harel, Idit (1991). *Situating Constructionism*, in "Constructionism", ISBN 0893917869 , Ablex Publishing Corporation, Norwood, NJ, USA. Referenced from the on-line version, retrieved from http://www.papert.org/articles/SituatingConstructionism.html on August 6th, 2004.

Plato Learning, Inc. (2004). *3 Decoders*, promotional video in digital format describing the software application on-line at http://www.plato.com/downloads/movies/3_DECODERS_300.mov, retrieved on July 30th, 2004.

Plato Learning, Inc. (sem data). *History*, Web page retrieved on July 30th, 2004, from http://www.plato.com/aboutus/company_history.asp.

Poole, Steven (2000). *Trigger Happy: Videogames and the Entertainment Revolution*, ISBN 1559705396, Arcade Publishing, New York, NY, USA.

Reiser, Robert A. (2001).*A History of Instructional Design and Technology: Part II: A History of Instructional Design*, in Educational Technology Research and Development, Vol. 49, No. 2, 2001, pp. 57–67, ISSN 1042–1629. Referenced from the electronic version, retrieved from http://www.aect.org/pdf/etr&d/4902/4902-04.pdf on June 30th, 2004.

Skinner, B. F. (1968). *The Technology of Teaching*, Prentice-Hall, Englewood Cliffs, NJ, USA. Referenced from the on-line version, retrieved on August 22nd, 2004, from http://www.cedu.niu.edu/tutortechlab/history/machine.html.

Smith, David Canfield; Cypher, Allen; Tesler, Larry (2001). *Novice Programming Comes of Age*, in Henry Lieberman (ed.), "Your Wish Is My Command: Programming By Example", ISBN 1-55860-688-2, ch. 1, pp. 7-19, Morgan Kaufmann Publishers, San Francisco, California, USA.

Solomon, Cynthia (1986). *Computers environments for children*, ISBN 0-262-19249-7, MIT Press, Cambridge, Massachusetts, USA.

Solomon, Cynthia (1996). *Origins of Educational Multimedia Environments*, in Allison Druin & Cynthia Solomon (eds.), "Designing Multimedia Environments for Children", ISBN 0-471-11688-2, ch. 1, John Wiley & Sons, Inc., New York, NY, USA.

Suppes, Patrick (1995). *The Aims of Education*, in Neiman, Alven; Curren, Randall R., Paul; McCarthy, Christine; Luise Prior, McCarty; Rice, Suzanne; Dummit, Diana; Duncan, Barbara (eds.), "Philosophy of Education Yearbook", ISBN 9995796589, Philosophy of Education Society, University of Illinois, Champaign, Illinois, USA. Referenced from the on-line version, retrieved on July 27th, 2004, from http://www.ed.uiuc.edu/EPS/PES-Yearbook/95_docs/suppes.html.

Thrall, Tony; Tingey, Barbara (2003). *SuccessMaker® Motion: A Research Summary*, Technical Note TN030409, Pearson Digital Learning, Pearson Education, Upper Saddle River, NJ, USA. Referenced from the on-line version, retrieved on June 30th, 2004, from http://www.pearsondigital.com/pdfs/whitepapers/SuccessMaker_Motion-A_Research_Summary.pdf

Turkle, Sherry and Papert, Seymour (1990). *Epistemological Pluralism: Styles and Voices within the Computer Culture*, Epistemology and Learning Group Memo no. 3, September 1990, Epistemology and Learning Group, Media Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA. Also published in "Signs", ISSN 0097-9740, vol. 16, no. 1, 1990, pp. 128-155, The University of Chicago Press, Chicago, Illinois, USA. Referenced from the on-line version,

retrieved on July 2nd, 2004, from www-personal.si.umich.edu/~rfrost/courses/Women+Tech/readings/Turkle%20(Signs).pdf.

U.S. Bureau of the Census (1988). *Computer Use in the United States: 1984*, U.S. Department of Commerce, Washington, D.C., USA. Referenced from the online version at http://www.census.gov/population/socdemo/computer/p23-155/p23-155.pdf, retrieved on August 4th, 2004.

U.S. Bureau of the Census (1993). *Statistical Abstract of the United States: 1993*, 113th edition, U.S. Department of Commerce, Washington, D.C., USA. Referenced from the online version, retrieved from http://www2.census.gov/prod2/statcomp/documents/1993-01.pdf and http://www2.census.gov/prod2/statcomp/documents/1993-05.pdf, on August 4th, 2004.

USCFC, U.S. Centennial of Flight Commision (sem data). *Commemorating a Century of Wings - An Overview*, Web page at http://www.centennialofflight.gov/essay/Wright_Bros/WR_OV.htm, last accessed on May 18th, 2005.

Vargas, Julie Skinner (sem data). *Brief biography of B.F. Skinner*, retrieved on June 30th, 2004, from the Web site of the B. F. Skinner Foundation, at http://www.bfskinner.org/bio.asp.

Viacom, Inc. (1998). *Viacom To Complete Sale Of Non-Consumer Publishing Operations To Pearson On November 27*, on-line press release, retrieved on July 23rd, 2004, from http://www.viacom.com/press.tin?ixPressRelease=40000904.

Wells, Herbert George (1937). *World Brain: The Idea of a Permanent World Encyclopaedia*, in "Encyclopédie Française", Société de gestion de l'Encyclopédie française, Paris, France. Referenced from the electronic version, retrieved from http://art-bin.com/art/obrain.html on July 1st, 2004.